Table of Contents

Burp Suite 实战指南

引子

刚接触web安全的时候，非常想找到一款集成型的渗透测试工具，找来找去，最终选择了Burp Suite，除了它功能强大之外，还有就是好用，易于上手。于是就从网上下载了一个破解版的来用，记得那时候好像是1.2版本，功能也没有现在这么强大。在使用的过程中，慢慢发现，网上系统全量的介绍BurpSuite的书籍太少了，大多是零星、片段的讲解，不成体系。后来慢慢地出现了不少介绍BurpSuite的视频，现状也变得越来越好。但每每遇到不知道的问题时，还是不得不搜寻BurpSuite的官方文档和英文网页来解决问题，也正是这些问题，慢慢让我觉得有必要整理一套全面的BurpSuite中文教程，算是为web安全界做尽自己的一份微薄之力，也才有了你们现在看到的这一系列文章。

我给这些文章取了IT行业图书比较通用的名称:《BurpSuite实战指南》，您可以称我为中文编写者，文章中的内容主要源于BurpSuite官方文档和多位国外安全大牛的经验总结，我只是在他们的基础上，结合我的经验、理解和实践，编写成现在的中文教程。本书我也没有出版成纸质图书的计划，本着IT人互联分享的精神，放在github，做免费的电子书。于业界，算一份小小的贡献；于自己，算一次总结和锻炼。

以上，是为小记。

感谢您阅读此书，阅读过程中，如果发现错误的地方，欢迎发送邮件到 t0data@hotmail.com，感谢您的批评指正。

本书包含以下章节内容：

第一部分 Burp Suite 基础

第二部分 Burp Suite 高级

第三部分 Burp Suite 综合使用

第一章 Burp Suite 安装和环境配置

Burp Suite是一个集成化的渗透测试工具，它集合了多种渗透测试组件，使我们自动化地或手工地能更好的完成对web应用的渗透测试和攻击。在渗透测试中，我们使用Burp Suite将使得测试工作变得更加容易和方便，即使在不需要娴熟的技巧的情况下，只有我们熟悉Burp Suite的使用，也使得渗透测试工作变得轻松和高效。

Burp Suite是由Java语言编写而成，而Java自身的跨平台性，使得软件的学习和使用更加方便。Burp Suite不像其他的自动化测试工具，它需要你手工的去配置一些参数，触发一些自动化流程，然后它才会开始工作。

Burp Suite可执行程序是java文件类型的jar文件，免费版的可以从免费版下载地址进行下载。免费版的Burp Suite会有许多限制，很多的高级工具无法使用，如果您想使用更多的高级功能，需要付费购买专业版。专业版与免费版的主要区别有

1. Burp Scanner
2. 工作空间的保存和恢复
3. 拓展工具，如Target Analyzer, Content Discovery和 Task Scheduler

本章主要讲述Burp Suite的基本配置，包含如下内容：

- 如何从命令行启动Burp Suite</br>
- 如何设置JVM内存 大小</br>
- IPv6问题调试

如何从命令行启动Burp Suite

Burp Suite是一个无需安装软件，下载完成后，直接从命令行启用即可。但Burp Suite是用Java语言开发的，运行时依赖于JRE，需要提前Java可运行环境。如果没有配置Java环境或者不知道如何配置的童鞋请参考win7电脑上的Java环境配置 配置完Java环境之后，首先验证Java配置是否正确，如果输入java -version 出现下图的结果，证明配置正确且已完成。

```
管理员: C:\Windows\system32\cmd.exe

C:\Users\      >java -version
java version "1.7.0_17"
Java(TM) SE Runtime Environment (build 1.7.0_17-b02)
Java HotSpot(TM) 64-Bit Server VM (build 23.7-b01, mixed mode)
```

这时，你只要在cmd里执行java -jar /your_burpsuite_path/burpSuite.jar即可启动Burp Suite,或者，你将Burp Suite的jar放入class_path目录下，直接执行java -jar burpSuite.jar也可以启动。

==注意：your_burpsuite_path为你Burp Suite所在路径，burpSuite.jar文件名必须跟你下载的jar文件名称一致==

如何设置JVM内存 大小

如果Java可运行环境配置正确的话，当你双击burpSuite.jar即可启动软件，这时，Burp Suite自己会自动分配最大的可用内存，具体实际分配了多少内存，默认一般为64M。当我们在渗透测试过程，如果有成千上万个请求通过Burp Suite，这时就可能会导致Burp Suite因内存不足而崩溃，从而会丢失渗透测试过程中的相关数据，这是我们不希望看到的。因此，当我们启动Burp Suite时，通常会指定它使用的内存大小。 一般来说，我们通常会分配2G的内存供Burp Suite使用，如果你的电脑内存足够，可以分配4G；如果你的电脑内存足够小，你也可以分配128M。当你给Burp Suite分配足够多的内存时，它能做的工作也会更多。指定Burp Suite占用内存大小的具体配置方法是在启动脚本里添加如下命令行参数： 假设启动脚本的名称为burp_suite_start.bat，则该bat脚本的内容为

```
java -jar -Xmx2048M  /your_burpsuite_path/burpsuite.jar
```

其中参数-Xmx指定JVM可用的最大内存，单位可以是M，也可以是G，如果是G为单位的话，则脚本内容为：

```
java -jar -Xmx2G  /your_burpsuite_path/burpsuite.jar
```

更多关于*JVM*性能调优的知识请阅读 *Oracle JVM Tuning*

IPv6问题调试

Burp Suite是不支持IPv6地址进行数据通信的，这时在cmd控制台里就会抛出如下异常

```
java.net.SocketException: Permission denied
```

同时，浏览器访问时，也会出现异常

```
Burp proxy error: Permission denied: connect
```

当出现如上问题时，我们需要修改启动脚本，添加对IPv4的指定后，重启Burp Suite即可。

```
java -jar -Xmx2048M  -Djava.net.preferIPv4Stack=true  /your_burpsuite_path/burpsuite.jar
```

通过 -Djava.net.preferIPv4Stack=true参数的设置，告诉Java运行环境，使用IPv4协议栈进行数据通信，IPv6协议将会被禁止使用。 这个错误最常见于64位的windows操作系统上，使用了32位的JDK

第二章 Burp Suite代理和浏览器设置

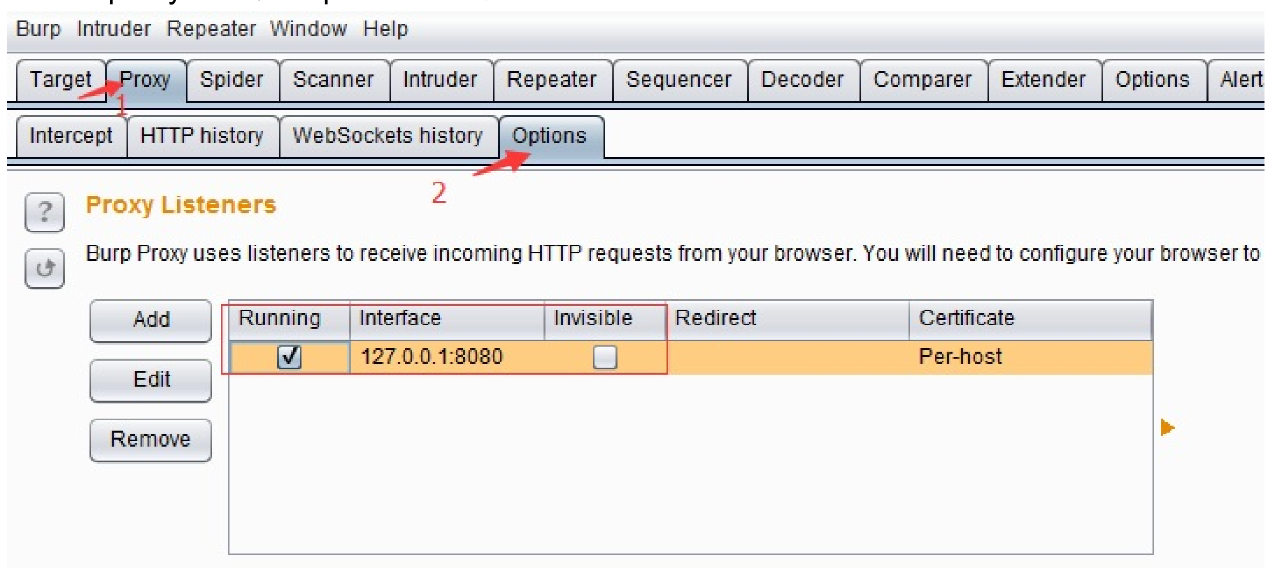
Burp Suite代理工具是以拦截代理的方式，拦截所有通过代理的网络流量，如客户端的请求数据、服务器端的返回信息等。Burp Suite主要拦截http和https协议的流量，通过拦截，Burp Suite以中间人的方式，可以对客户端请求数据、服务端返回做各种处理，以达到安全评估测试的目的。

在日常工作中，我们最常用的web客户端就是的web浏览器，我们可以通过代理的设置，做到对web浏览器的流量拦截，并对经过Burp Suite代理的流量数据进行处理。

下面我们就分别看看IE、Firefox、Google Chrome下是如何配置Burp Suite代理的。

IE设置

当Burp Suite 启动之后，默认分配的代理地址和端口是127.0.0.1 ：8080,我们可以从Burp Suite的proxy选项卡的options上查看。如图：

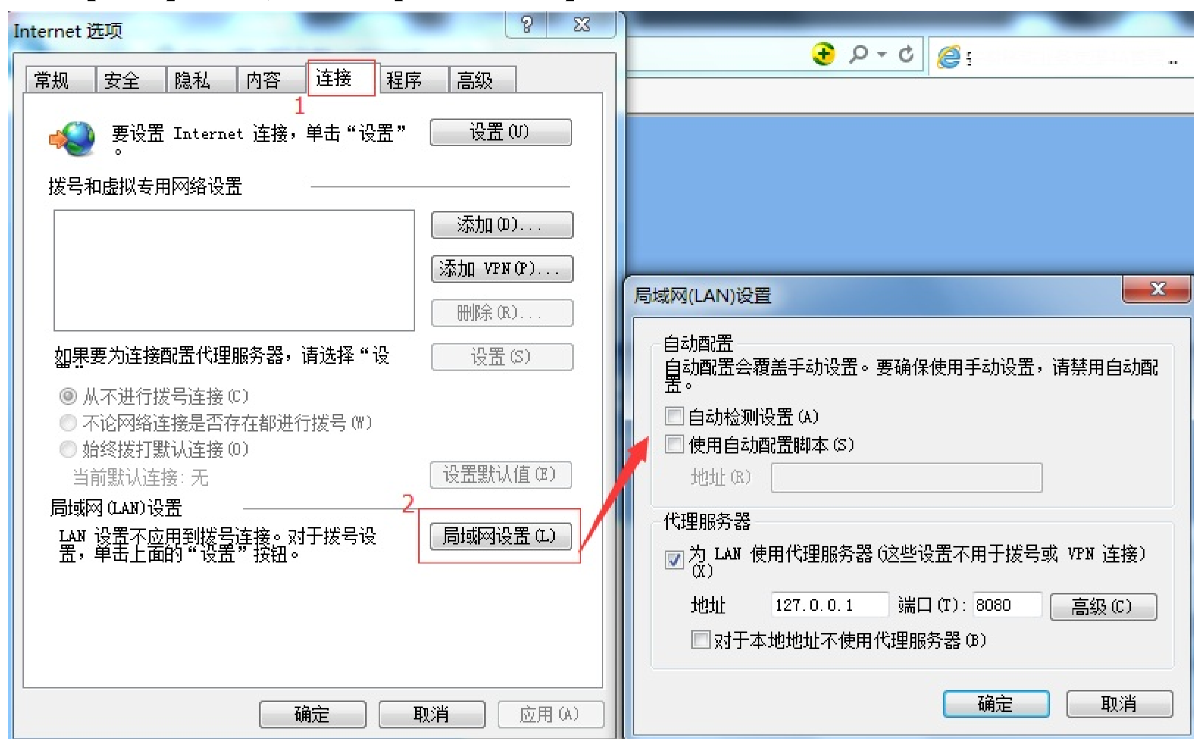


现在，我们通过如下步骤的设置即可完成IE通过Burp Suite 代理的相关配置。

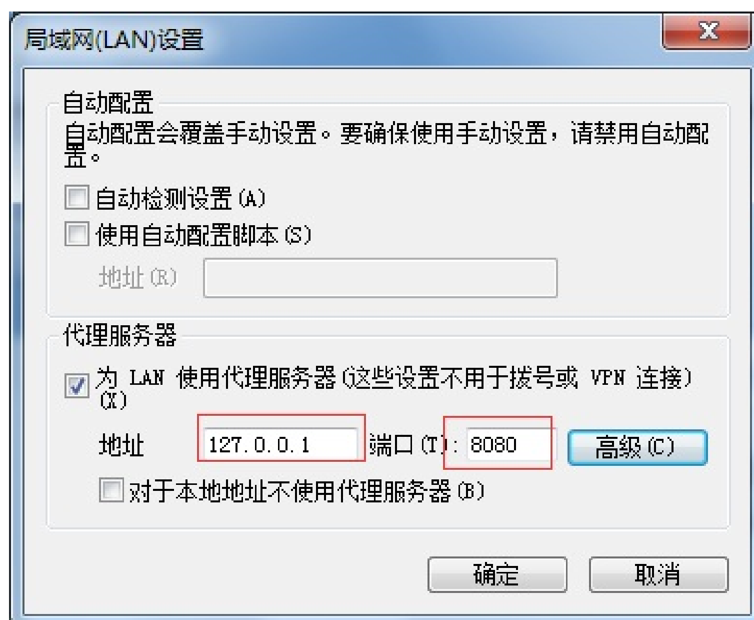
1. 启动IE浏览器
2. 点击【工具】菜单，选择【Internet】选项



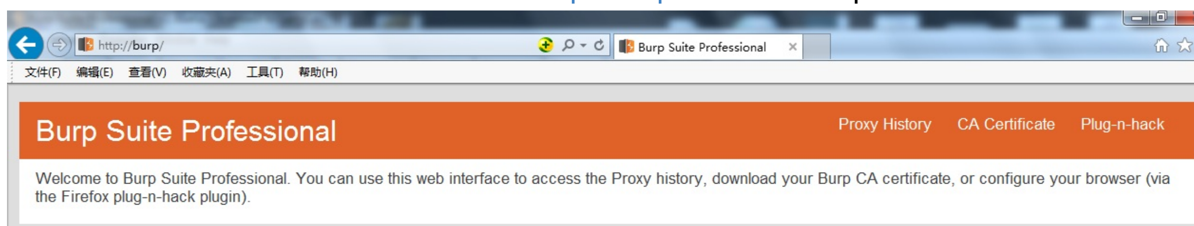
3. 打开【连接】选项卡，点击【局域网设置】，进行代理设置。



4. 在代理服务器设置的地址输入框中填写127.0.0.1,端口填写8080，点击【确定】，完成代

理服务器的设置。

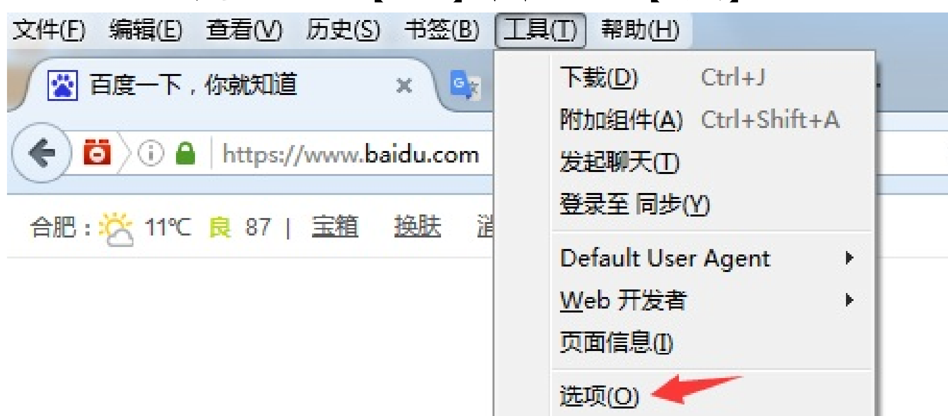
5. 这时，IE的设置已经完成，你可以访问 http://burp 将会看到Burp Suite的欢迎界面。



FireFox设置

与IE的设置类似，在FireFox中，我们也要进行一些参数设置，才能将FireFox浏览器的通信流量，通过Burp Suite代理进行传输。详细的步骤如下：

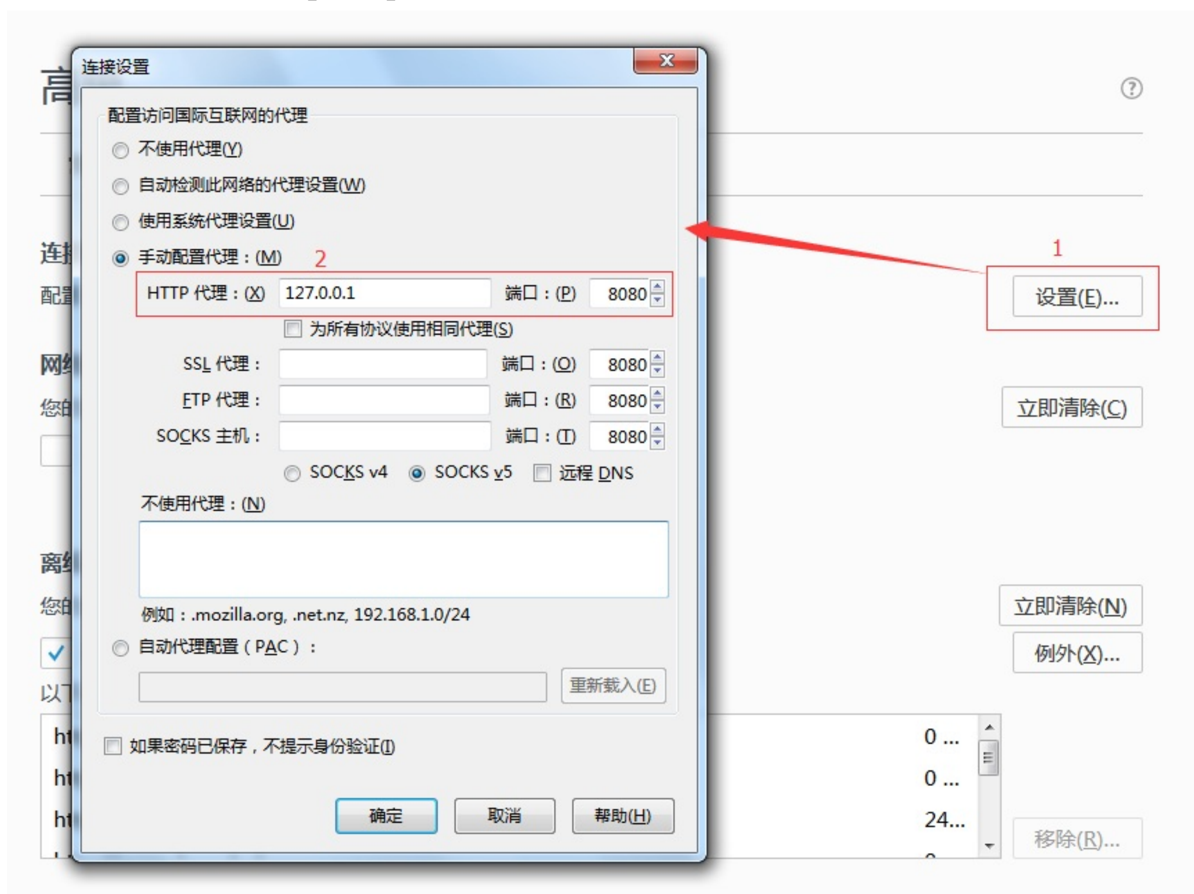
1. 启动FireFox浏览器，点击【工具】菜单，点击【选项】。



2. 在新打开的about:preferences#advanced窗口中，依次点击【高级】-【网络】，我们将会看到FireFox连接网络的设置选项。



3. 点击【设置】，在弹出的【连接设置】对话框中，找到“http代理”，填写127.0.0.1，端口填写8080，最后点击【确认】保存参数设置，完成FireFox的代理配置。



当然，FireFox浏览器中，可以添加FireFox的扩展组件，对代理服务器进行管理。例如FireX Proxy、Proxy Swither都是很好用的组件，感兴趣的读者可以自己下载试用一下。

Google Chrome设置

Google Chrome使用Burp Suite作为代理服务器的配置步骤如下：

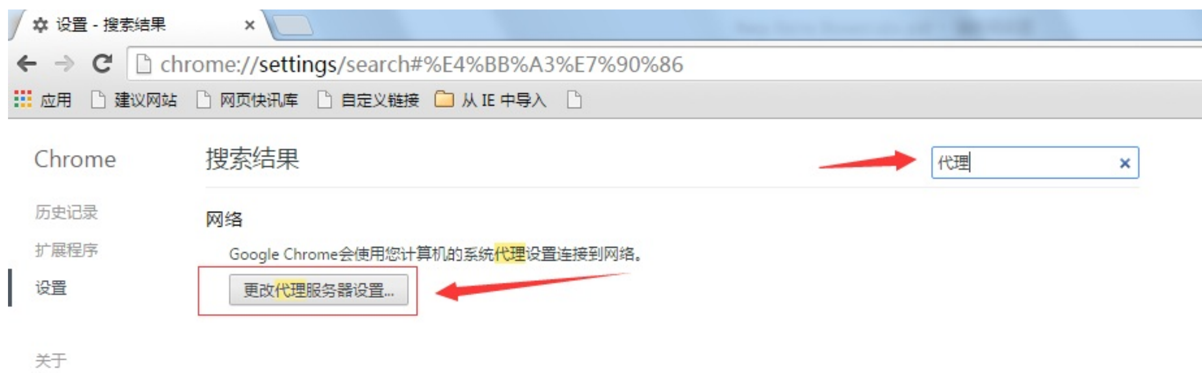
1. 启动Google Chrome浏览器，在地址栏输入chrome://settings/，回车后即显示Google Chrome浏览器的配置界面



2. 点击底部的【显示高级设置】，将显示Google Chrome浏览器的高级设置。



3. 当然，你也可以直接在搜索框中输入"代理"，回车后将自动定位到代理服务器设置功能。



4. 点击【更改代理服务器设置】，windows系统下将会弹出IE浏览器的代理设置，此时，按照IE浏览器的设置步骤，完成代理服务器的配置即可。

除了上述的三种常用的浏览器外，还有Safari浏览器也有不少的用户在使用，其代理配置请点击阅读进行查看。

第三章 如何使用Burp Suite代理

Burp Proxy 是Burp Suite以用户驱动测试流程功能的核心，通过代理模式，可以让我们拦截、查看、修改所有在客户端和服务端之间传输的数据。

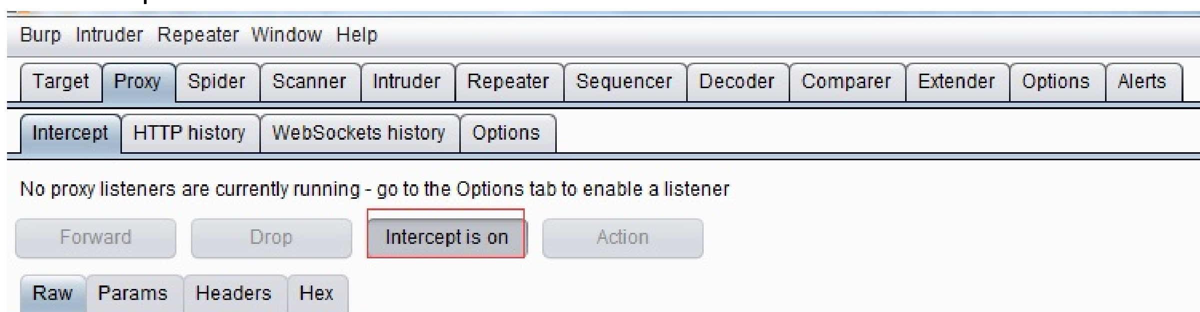
本章主要讲述以下内容：

- Burp Proxy基本使用
- 数据拦截与控制
- 可选项配置Options
- 历史记录History

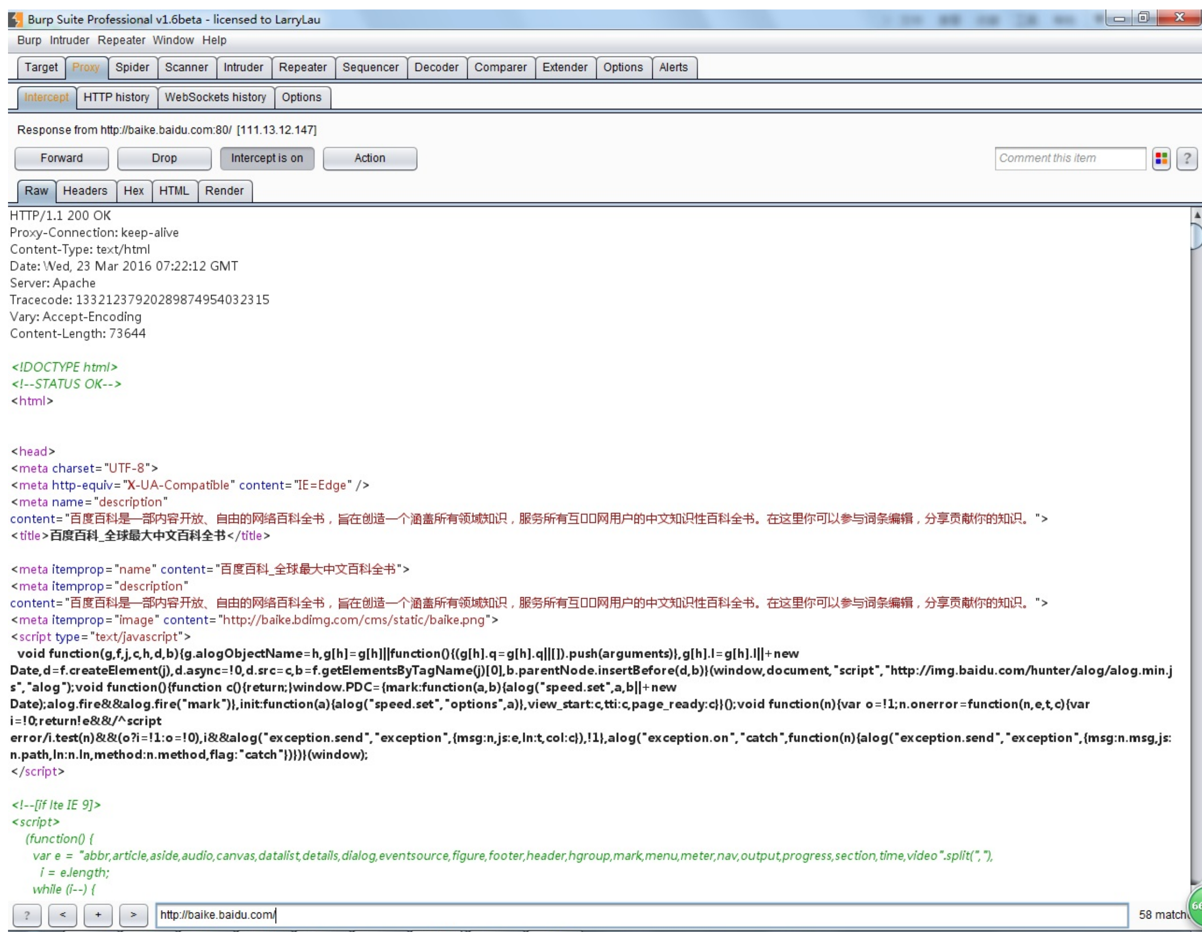
Burp Proxy基本使用

通过上一章的学习，我们对Burp Suite代理模式和浏览器代理设置有了基本的了解。Burp Proxy的使用是一个循序渐进的过程，刚开始使用时，可能并不能很快就获取你所期望的结果，慢慢地当你熟悉了它的功能和使用方法，你就可以用它很好地对一个产品系统做安全能力评估。 一般使用Burp Proxy时，大体涉及环节如下：

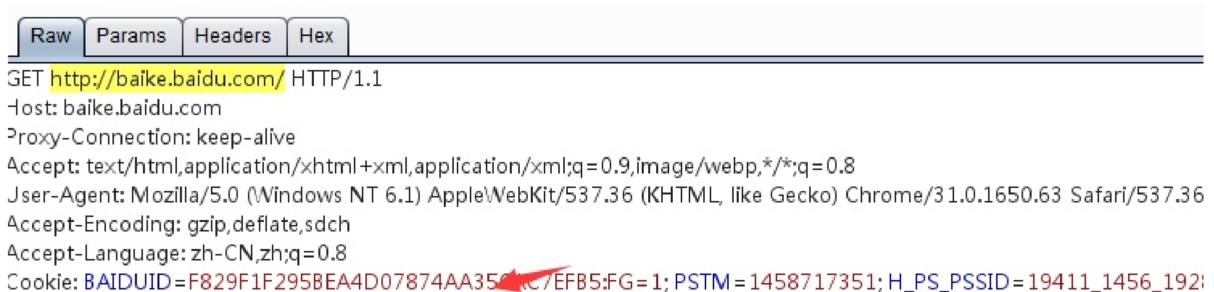
1. 首先，确认JRE已经安装好，Burp Suite可以启动并正常运行，且已经完成浏览器的代理服务器配置。
2. 打开Proxy功能中的Intercept选项卡，确认拦截功能为“Interception is on”状态，如果显示为“Intercept is off”则点击它，打开拦截功能。



3. 打开浏览器，输入你需要访问的URL（以http://baike.baidu.com/为例）并回车，这时你将会看到数据流量经过Burp Proxy并暂停，直到你点击【Forward】，才会继续传输下去。如果你点击了【Drop】，则这次通过的数据将会被丢失，不再继续处理。
4. 当我们点击【Forward】之后，我们将看到这次请求返回的所有数据。



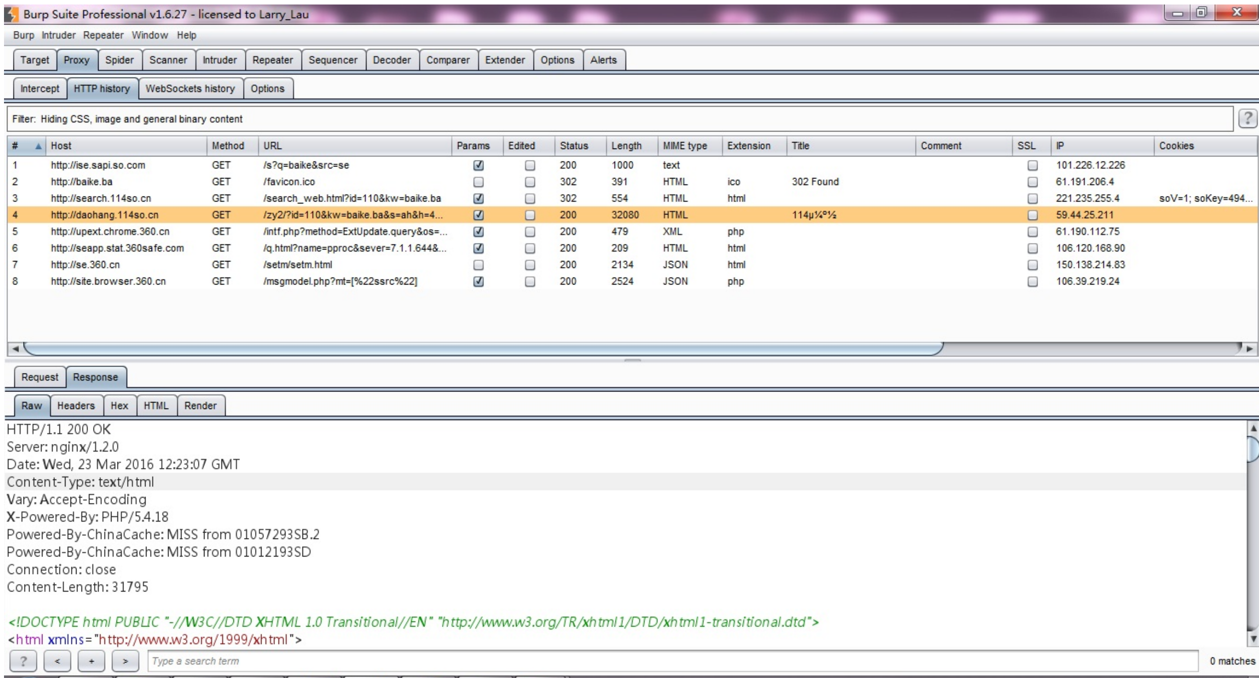
5. 当Burp Suite拦截的客户端和服务器交互之后，我们可以在Burp Suite的消息分析选项卡中查看这次请求的实体内容、消息头、请求参数等信息。消息分析选项视图主要包括以下四项：



6. Raw 这是视图主要显示web请求的raw格式，包含请求地址、http协议版本、主机头、浏览器信息、Accept可接受的内容类型、字符集、编码方式、cookie等。你可以通过手工修改这些信息，对服务器端进行渗透测试。
7. params 这个视图主要显示客户端请求的参数信息、包括GET或者POST请求的参数、Cookie参数。渗透人员可以通过修改这些请求参数来完成对服务器端的渗透测试。
8. headers 这个视图显示的信息和Raw的信息类似，只不过在这个视图中，展示得更直观、友好。
9. Hex 这个视图显示Raw的二进制内容，你可以通过hex编辑器对请求的内容进行修改。

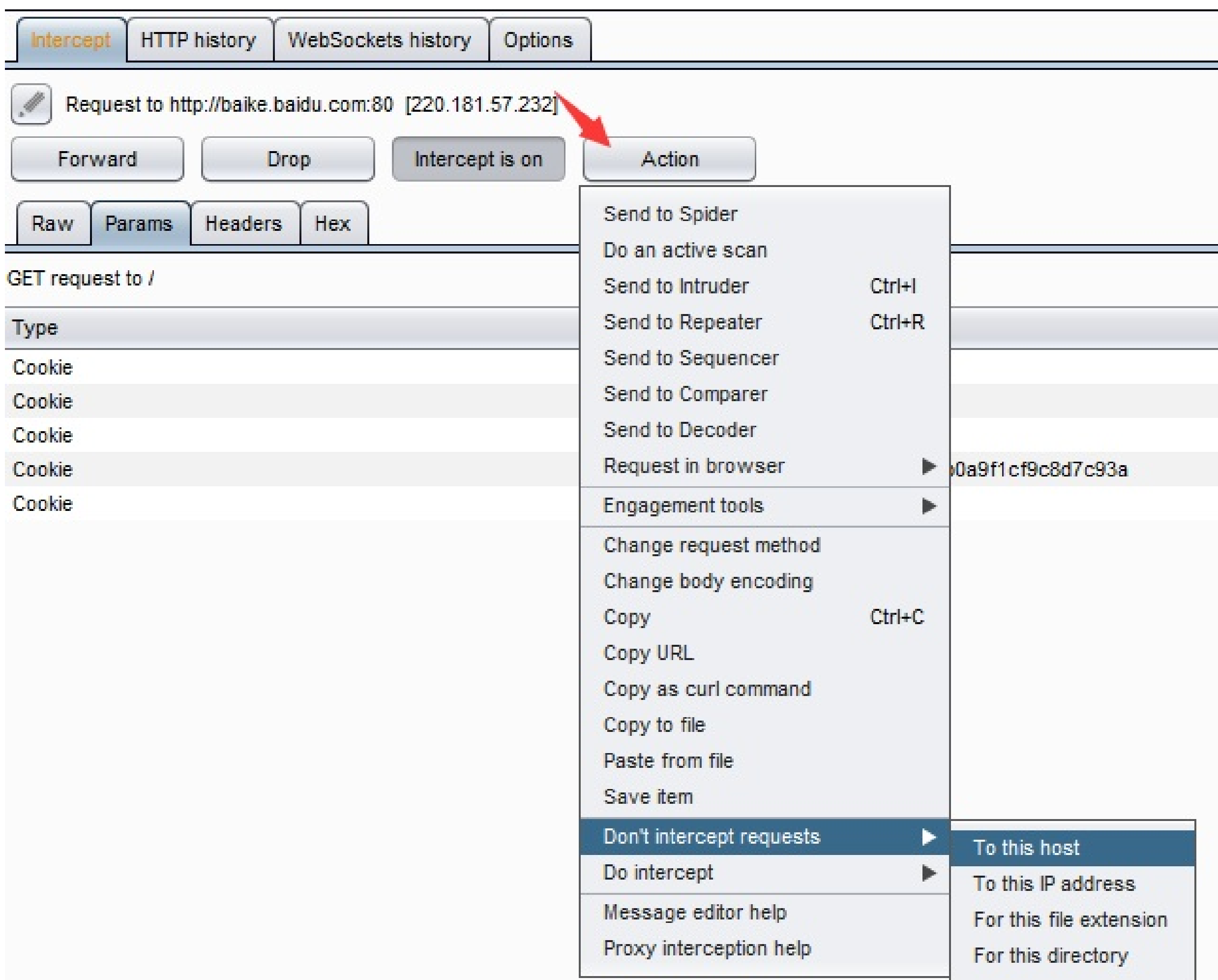
默认情况下，Burp Proxy只拦截请求的消息，普通文件请求如css、js、图片是不会被拦截的，你可以修改默认的拦截选项来拦截这些静态文件，当然，你也可以通过修改拦截的作用域、参数或者服务器端返回的关键字来控制Burp Proxy的消息拦截，这些在后面的章节中我

们会进一步的学习。 所有流经Burp Proxy的消息，都会在http history记录下来，我们可以通过历史选项卡，查看传输的数据内容，对交互的数据进行测试和验证。同时，对于拦截到的消息和历史消息，都可以通过右击弹出菜单，发送到Burp的其他组件，如Spider、Scanner、Repeater、Intruder、Sequencer、Decoder、Comparer、Extender，进行进一步的测试。如下图所示：

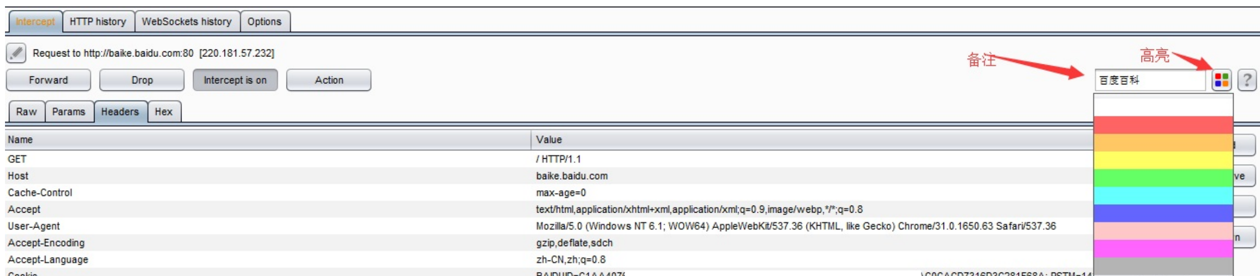


数据拦截与控制

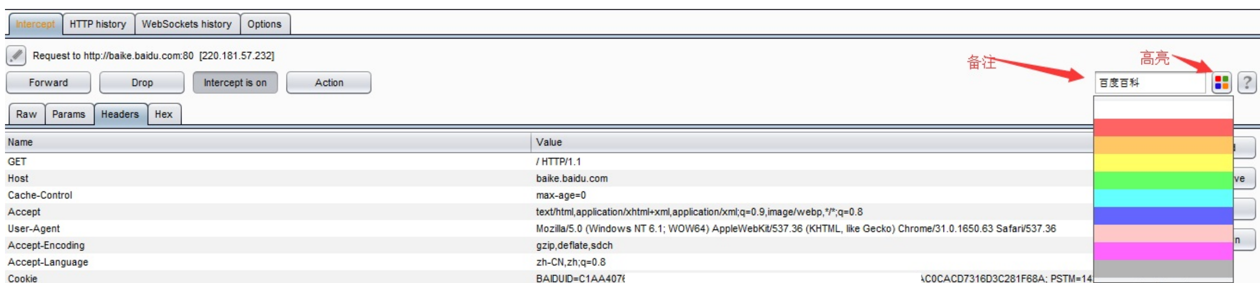
Burp Proxy的拦截功能主要由Intercept选项卡中的Forward、Drop、Interception is on/off、Action、Comment 以及Highlight构成，它们的功能分别是： **Forward**的功能是当你查看过消息或者重新编辑过消息之后，点击此按钮，将发送消息至服务器端。 **Drop**的功能是你想丢失当前拦截的消息，不再forward到服务器端。 **Interception is on**表示拦截功能打开，拦截所有通过Burp Proxy的请求数据；**Interception is off**表示拦截功能关闭，不再拦截通过Burp Proxy的所有请求数据。 **Action**的功能是除了将当前请求的消息传递到Spider、Scanner、Repeater、Intruder、Sequencer、Decoder、Comparer组件外，还可以做一些请求消息的修改，如改变GET或者POST请求方式、改变请求body的编码，同时也可以改变请求消息的拦截设置，如不再拦截此主机的消息、不再拦截此IP地址的消息、不再拦截此种文件类型的消息、不再拦截此目录的消息，也可以指定针对此消息拦截它的服务器端返回消息。



Comment的功能是指对拦截的消息添加备注，在一次渗透测试中，你通常会遇到一连串的请求消息，为了便于区分，在某个关键的请求消息上，你可以添加备注信息。



Highlight的功能与Comment功能有点类似，即对当前拦截的消息设置高亮，以便于其他的请求消息相区分。



除了Intercept中可以对通过Proxy的消息进行控制外，在可选项设置选项卡Options中也有很多的功能设置也可以对流经的消息进行控制和处理。

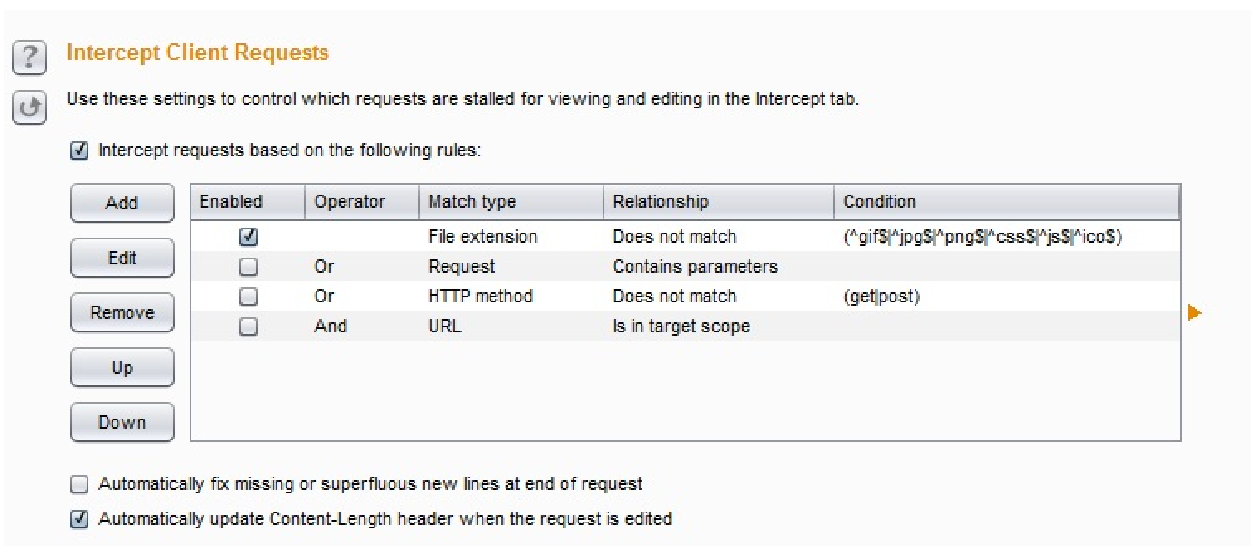
可选项配置Options

当我们打开可选项设置选项卡Options，从界面显示来看，主要包括以下几大板块（涉及https的功能不包含在本章内容里，后面会一章专门叙述）：

- 客户端请求消息拦截
- 服务器端返回消息拦截
- 服务器返回消息修改
- 正则表达式配置
- 其他配置项

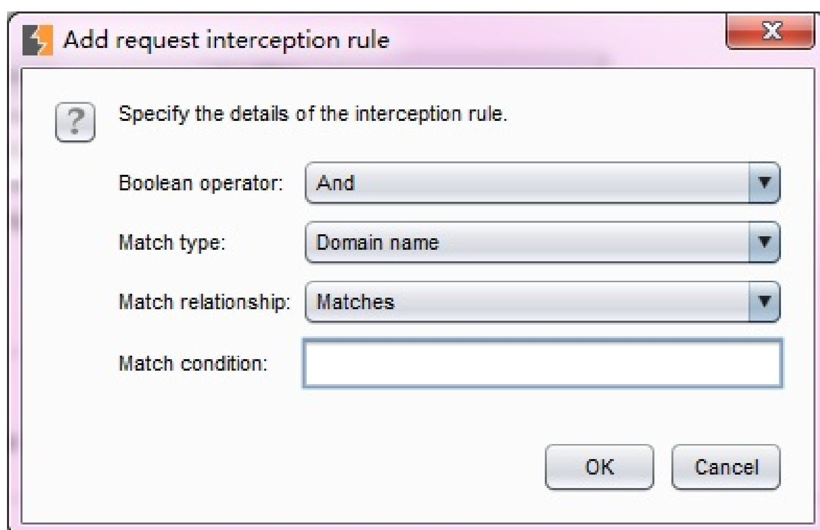
客户端请求消息拦截

客户端请求消息拦截是指拦截客户端发送到服务器端消息的相关配置选项，其界面如下：



主要包含拦截规则配置、错误消息自动修复、自动更新Content-Length消息头三个部分。

1. 如果intercept request based on the follow rules的checkbox被选中，则拦截所有符合勾选按钮下方列表中的请求规则的消息都将被拦截，拦截时，对规则的过滤是自上而下进行的。当然，我们可以根据自己的需求，通过【Up】和【Down】按钮，调节规则所在位置和排序。同时，我们可以点击【Add】添加一条规则，也可以选中一条规则，通过点击【Edit】进行编辑、点击【Remove】进行删除。当我们点击【Add】按钮时，会弹出规则添加的输入对话框，如下图：

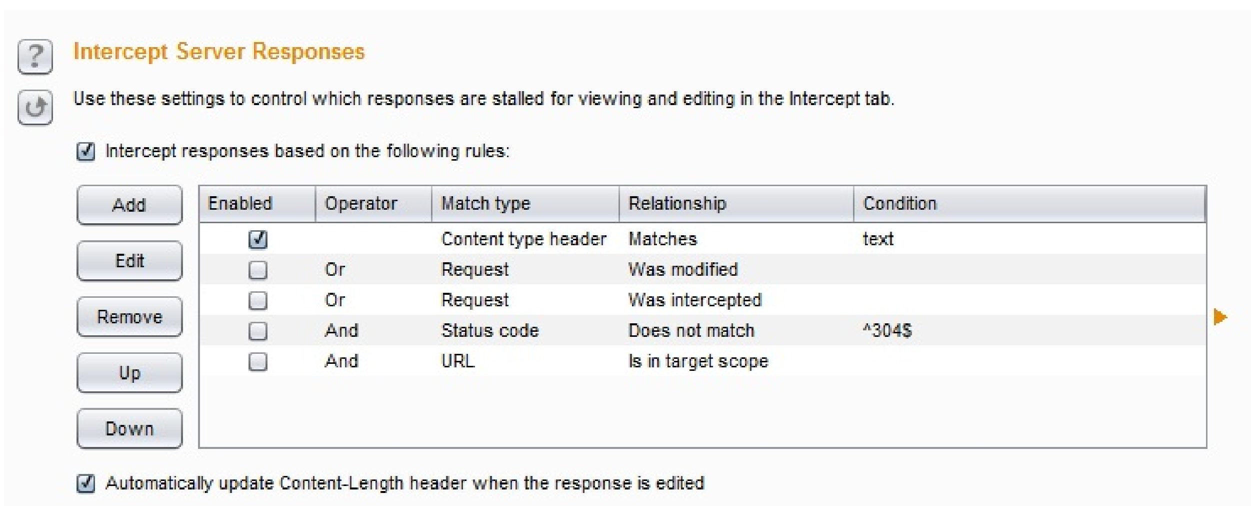


拦截规则添加时，共包含4个输入项。Boolean opertor表示当前的规则与其他规则是与的方式（And）还是或的方式（Or）共存；Match type表示匹配类型，此处匹配类型可以基于域名、IP地址、协议、请求方法、URL、文件类型、参数, cookies, 头部或者内容, 状态码, MIME类型, HTML页面的title等。Match relationship表示此条规则是匹配还是不匹配Match condition输入的关键字。当我们输入这些信息，点击【OK】按钮，则规则即被保存。

2. 如果Automatically fix missing的checkbox被选中，则表示在一次消息传输中，Burp Suite会自动修复丢失或多余的新行。比如说，一条被修改过的请求消息，如果丢失了头部结束的空行，Burp Suite会自动添加上；如果一次请求的消息体中，URI编码参数中包含任何新的换行，Burp Suite将会移除。此项功能在手工修改请求消息时，为了防止错误，有很好的保护效果。
3. 如果Automatically update Content-Length的checkbox被选中，则当请求的消息被修改后，Content-Length消息头部也会自动被修改，替换为与之相对应的值。

服务器端返回消息拦截

服务器端返回消息拦截顾名思义是指拦截服务器端返回的消息的相关配置项，其界面如下：

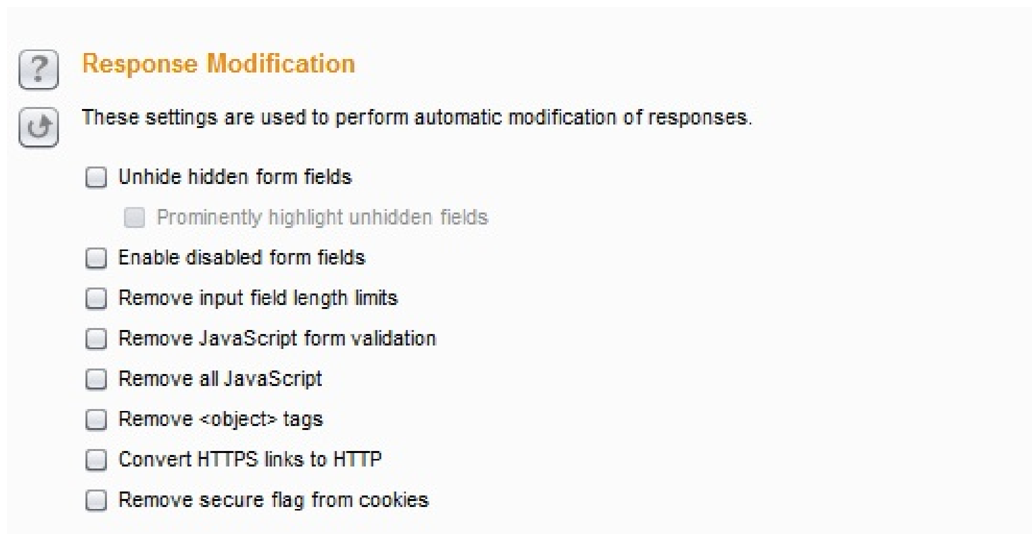


它的功能主要包含intercept response based on the follow rules和Automatically update

Content-Length header when the response edited两个选项，其功能分别与客户端请求消息拦截中的intercept request based on the follow rules、Automatically update Content-Length header when the request edited相对应，就不在赘述，请参上一节的内容。

服务器返回消息修改

服务器返回消息修改是指自动修改服务器端返回消息的相关设置项。其界面如下：



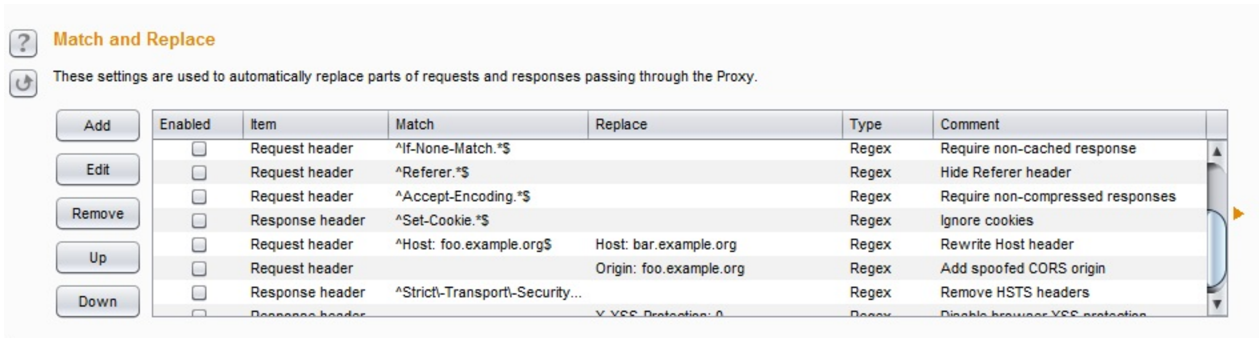
自上而下，每一个选择项分别对应的功能是

- 显示form表单中隐藏字段
- 高亮显示form表单中隐藏字段
- 使form表单中的disable字段生效，变成可输入域
- 移除输入域长度限制
- 移动JavaScript验证
- 移动所有的JavaScript
- 移除标签
- 转换https超链接为http链接
- 移除所有cookie中的安全标志

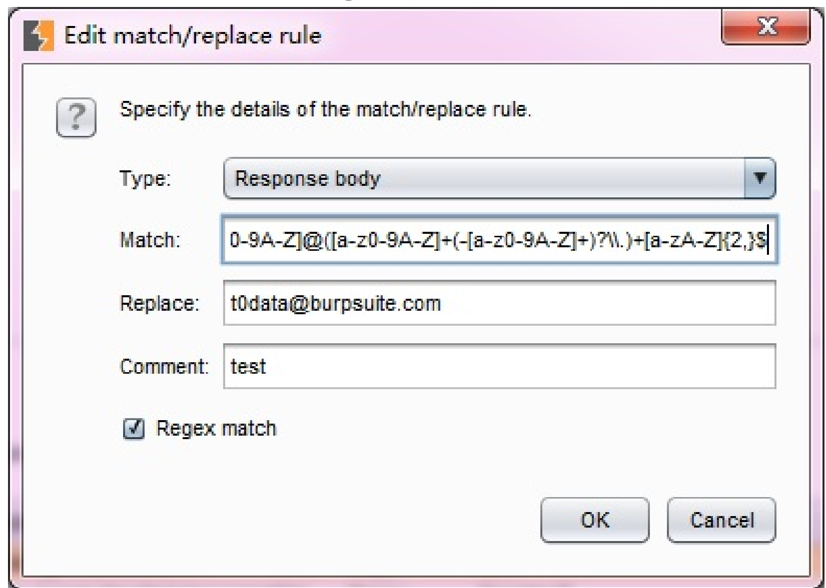
通过服务器返回消息修改可选择项的设置，可以方便渗透测试人员在安全评估过程中突破原有的数据限制，更好、更快地检测服务器端的安全性。

正则表达式配置

此项配置主要用来自动替换请求消息和服务器端返回消息中的某些值和文本，它与前文的规则的不同之处还在于支持正则表达式语言。



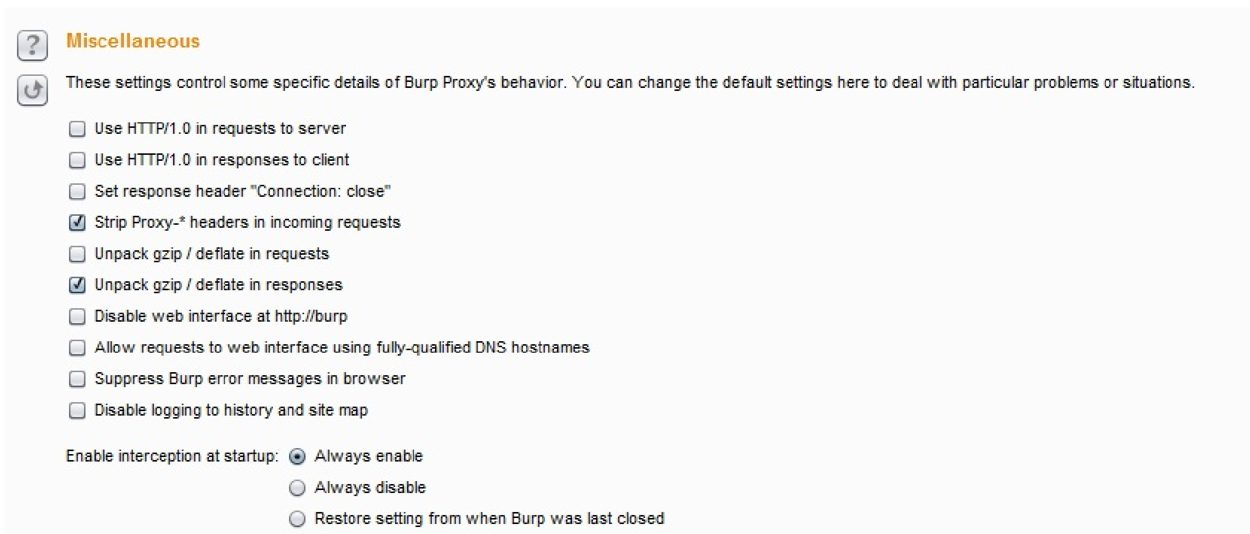
当点击【Add】按钮时，在弹出的匹配或替换规则输入对话框中我们可以看到，它可以对请求和返回消息的消息头，消息体、请求参数名、请求参数值、请求的第一行进行匹配和替换。例如，当我们要替换所有返回消息中的邮箱地址为t0data@burpsuite.com时，可以参考下图



的设置填写输入项并保存验证。

其他配置项

其他配置项主要是杂项设置。其界面如下：



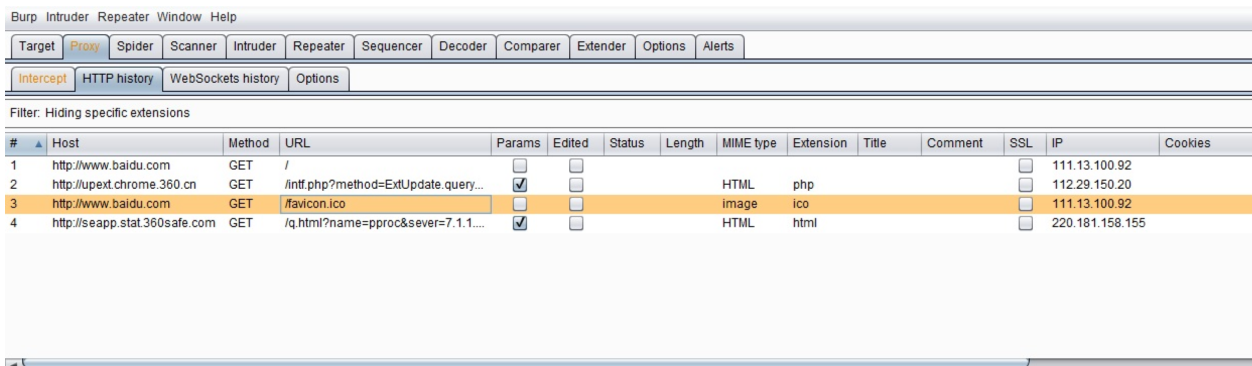
自上而下依次的功能是

- 指定使用HTTP/1.0协议与服务器进行通信 这项设置用于强制客户端采用HTTP/1.0协议与服务器进行通信，一般客户端使用的HTTP协议版本依赖于客户端浏览器，但某些服务器或者应用，必须使用HTTP/1.0协议，此时可勾选此项
- 指定使用HTTP/1.0协议反馈消息给客户端 目前所有的浏览器均支持HTTP/1.0协议和HTTP/1.1协议，强制指定HTTP/1.0协议主要用于显示浏览器的某些方面的特征，比如，阻止HTTP管道攻击。
- 设置返回消息头中的“Connection：close” 可用于某些情况下的阻止HTTP管道攻击。
- 请求消息头中脱掉Proxy-* 浏览器请求消息中，通常会携带代理服务器的相关信息，此选项主要用于清除消息头中的代理服务器信息。
- 解压请求消息中的压缩文件 某些应用在与服务器端进行交互时，会压缩消息体，勾选此选项，则Burp Suite 会自动解压消息体
- 解压返回消息中的压缩文件 大多数浏览器支持压缩的消息体，勾选此选项，则Burp Suite 会自动解压被服务器端压缩的消息体
- 禁用http://burp
- 允许通过DNS和主机名访问web接口 即允许通过域名或主机名访问Burp Suite
- 不在浏览器中显示Burp Suite错误 在我们使用Burp Suite时，如果发生了Burp Suite自身的错误，会在浏览器中显示，如果勾选了此项，则不会在浏览器中显示此类错误。
- 禁用日志到历史和网站地图中 此选项的作用是阻止记录日志到历史和网站地图，在某些情况下可能有用，比如说，通过上游服务器进行认证或者做正则表达式替换时，为了降低内存的消耗，减少日志的储存，你可以勾选此项。

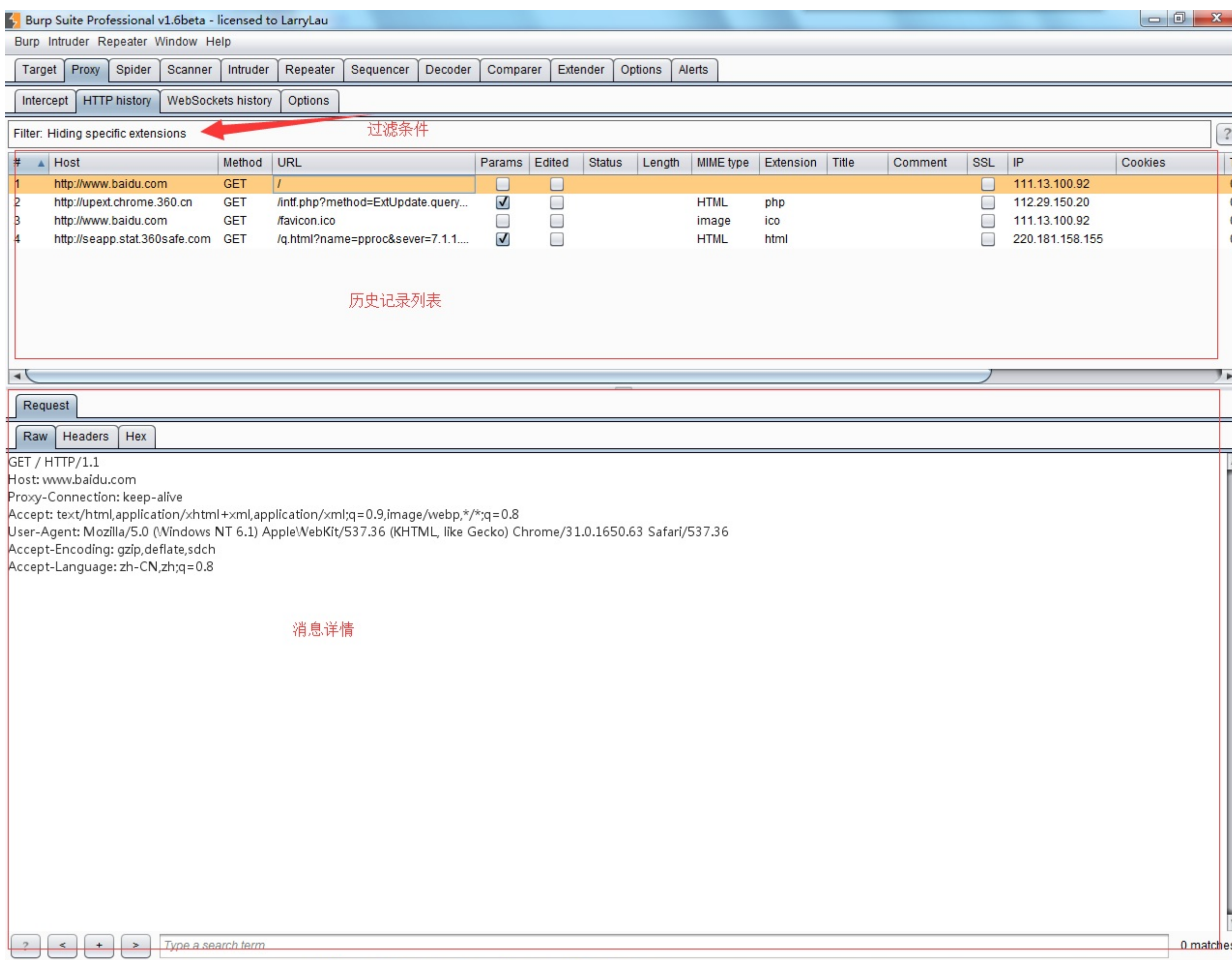
- 拦截功能开始设置
  这个选项主要用来配置intercept功能的生效方式，分为总是生效、 总是失效 、从上一次的Burp Suite中恢复设置3种方式。

历史记录History

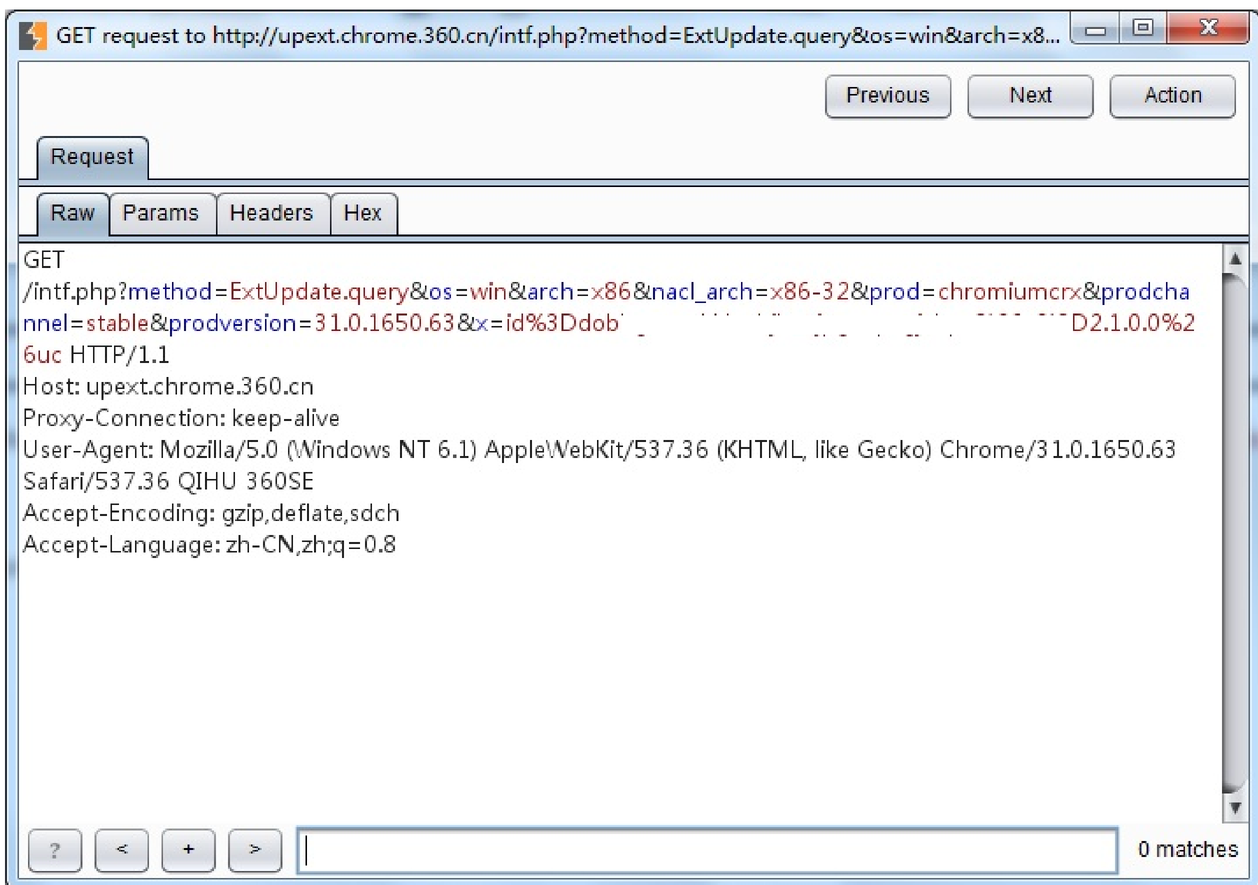
Burp Proxy的历史记录由HTTP历史和WebSockets历史两个部分组成。



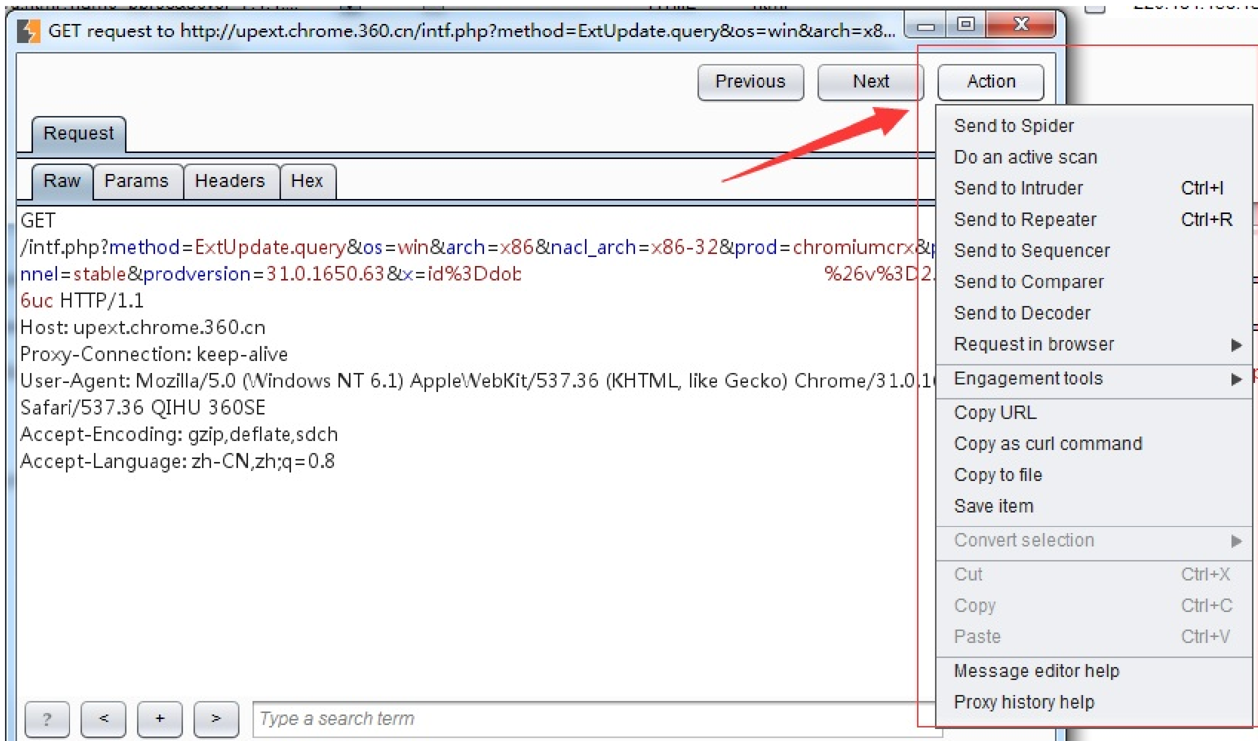
HTTP历史界面由筛选过滤器、历史记录列表、消息详情3个部分组成。



当我们在某一条历史记录上单击，会在下方的消息详解块显示此条消息的文本详细信息。当我们在某条消息上双击，则会弹出此条消息的详细对话框。

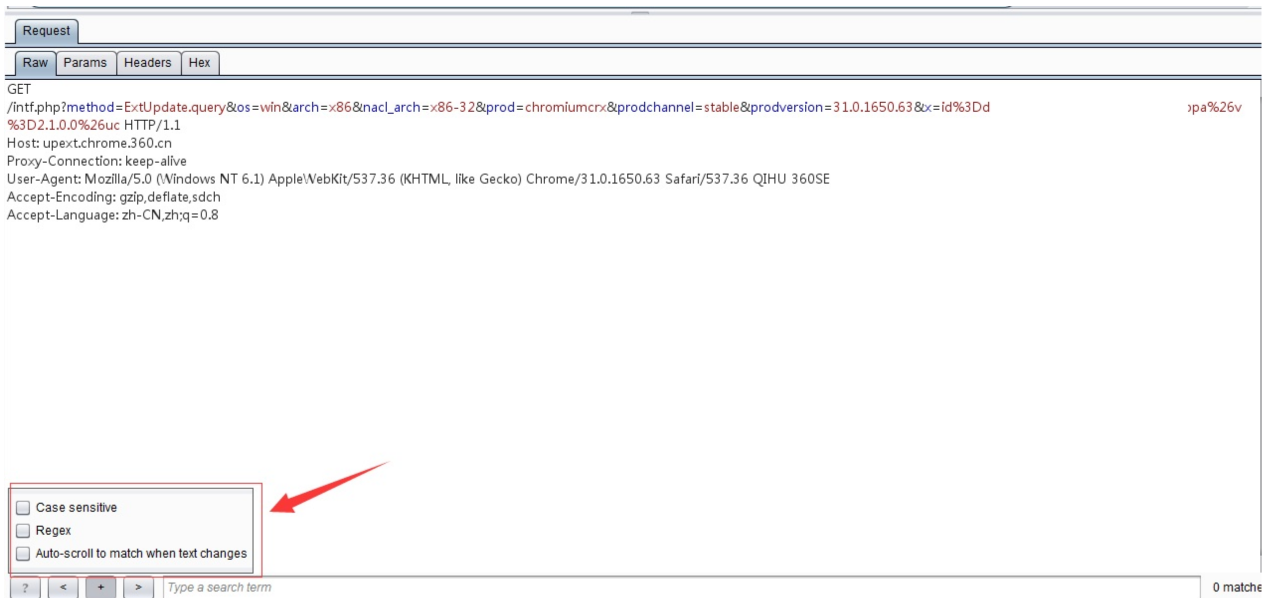


我们可以点击对话框右上方的【Previous】、【Next】按钮，浏览上一条或下一条消息的内容，也可以修改Raw的请求参数，然后执行多种【Action】操作。

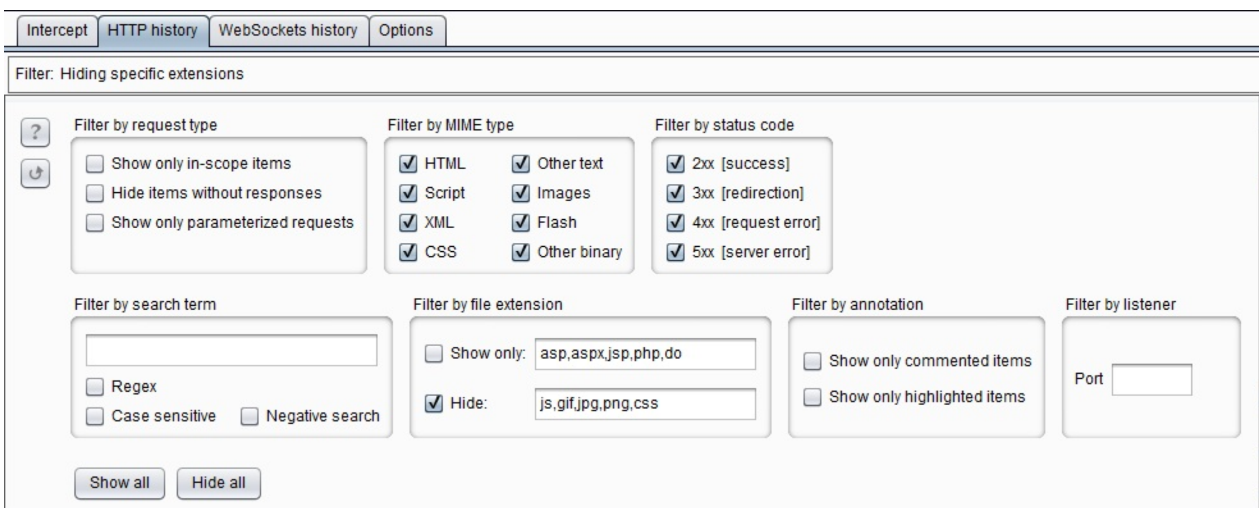


历史消息列表中主要包含请求序列号、请求协议和主机名、请求的方式、URL路径、请求参数、Cookie、是否用户编辑过消息、服务器端返回的HTTP状态码等信息。通过这些信息，我们可以对一次客户端与服务器端交互的HTTP消息详情做出准确的分析，同时，在下方的详情

视图中，也提供基于正则表达式方式的匹配查找功能，更好的方便渗透测试人员查找消息体中的相关信息。



当我们在做产品系统的安全评估过程中，会在HTTP历史中保存了大量的日志记录，为了更友好的消息管理，Burp提供了筛选过滤器功能。当我们点击HTTP历史标签下发的Filter时，将弹出筛选过滤器界面。



按照过滤条件的不同，筛选过滤器划分出7个子板块，分别是

- 按照请求类型过滤 你可以选择仅显示当前作用域的、仅显示有服务器端响应的和仅显示带有请求参数的消息。当你勾选“仅显示当前作用域”时，此作用域需要在Burp Target的Scope选项中进行配置，详细请阅读Burp Target相关章节。

- 按照MIME类型过滤 你可以控制是否显示服务器端返回的不同的文件类型的消息，比如只显示HTML、css或者图片。此过滤器目前支持HTML、Script、XML、CSS、其他文本、图片、Flash、二进制文件 8种形式。

- 按照服务器返回的HTTP状态码过滤 Burp根据服务器的状态码，按照2XX,3XX,4XX,5XX分别进行过滤。比如，如果你只想显示返回状态码为200的请求成功消息，则勾选2XX。

- 按照查找条件过滤 此过滤器是针对服务器端返回的消息内容，与输入的关键字进行匹配，具体的匹配方式，你可以选择 1.正则表达式 2.大小写敏感 3.否定查找 3种方式的任何组合，前面两种匹配方式容易理解，第3种匹配方式是指与关键字匹配上的将不再显示。

- 按照文件类型过滤 通过文件类型在过滤消息列表，这里有两个选择可供操作。一是仅仅显示哪些，另一个是不显示哪些。如果是仅仅显示哪些，在show only的输入框中填写显示的文件类型，同样，如果不显示哪些文件类型，只要在hide的输入框中填写不需要显示的文件类型即可。

- 按照注解过滤 此过滤器的功能是指，根据每一个消息拦截时候的备注或者是否高亮来作为筛选条件控制哪些消息在历史列表中显示。

- 按照监听端口过滤 此过滤器通常使用于当我们在Proxy Listeners中多个监听端口时，仅仅显示某个监听端口通信的消息，一般情况下，我们很少用到。

现在，我们再看看WebSockets历史选项的功能，从界面上我们可以看出，WebSockets历史所提供的功能和选项是HTTP历史的一个子集，只是因为采用的通信方式的不同，而被独立出来成为一个专门的视图。其功能的使用方式与HTTP历史雷同，此处就不在赘述。

通过本章的学习，你对Burp Suite的代理模式有了更深入的理解，知道了作为中间人的Burp Proxy在消息拦截过程中，可以对请求消息、应答消息做多方面的修改，并可以把消息传递给Burp的其他组件做进一步的测试。同时，Burp Proxy的历史日志功能和多种筛选过滤器让我们在使用中，能快速地查找需要的数据和关键信息，这些，都极大地帮助你提高了工作效率。

第四章 SSL和Proxy高级选项

在前一章，我们已经学习了HTTP消息如何通过Burp Proxy进行拦截和处理，本章我们将继续学习HTTPS协议消息的拦截和处理。

HTTPS协议是为了数据传输安全的需要，在HTTP原有的基础上，加入了安全套接字层SSL协议，通过CA证书来验证服务器的身份，并对通信消息进行加密。基于HTTPS协议这些特性，我们在使用Burp Proxy代理时，需要增加更多的设置，才能拦截HTTPS的消息。

本章包含的主要内容有

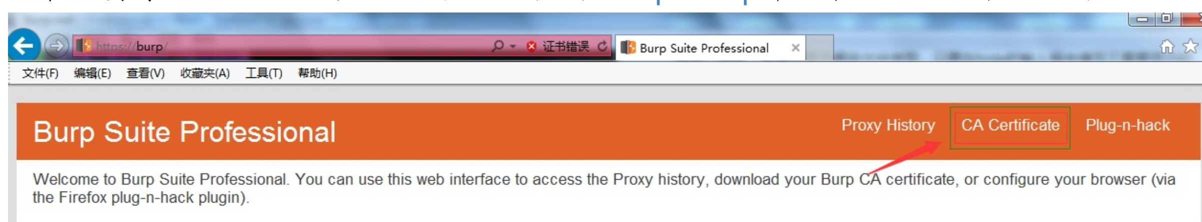
- CA证书的安装
- CA证书的卸载
- Proxy监听设置
- SSL直连和隐形代理设置

我们都知道，在HTTPS通信过程中，一个很重要的介质是CA证书，下面就我们一起来看看Burp Suite中CA证书的安装。

CA证书的安装

一般来说，Burp Proxy代理过程中的CA主要分为如下几个步骤（以win7下IE9为例）：

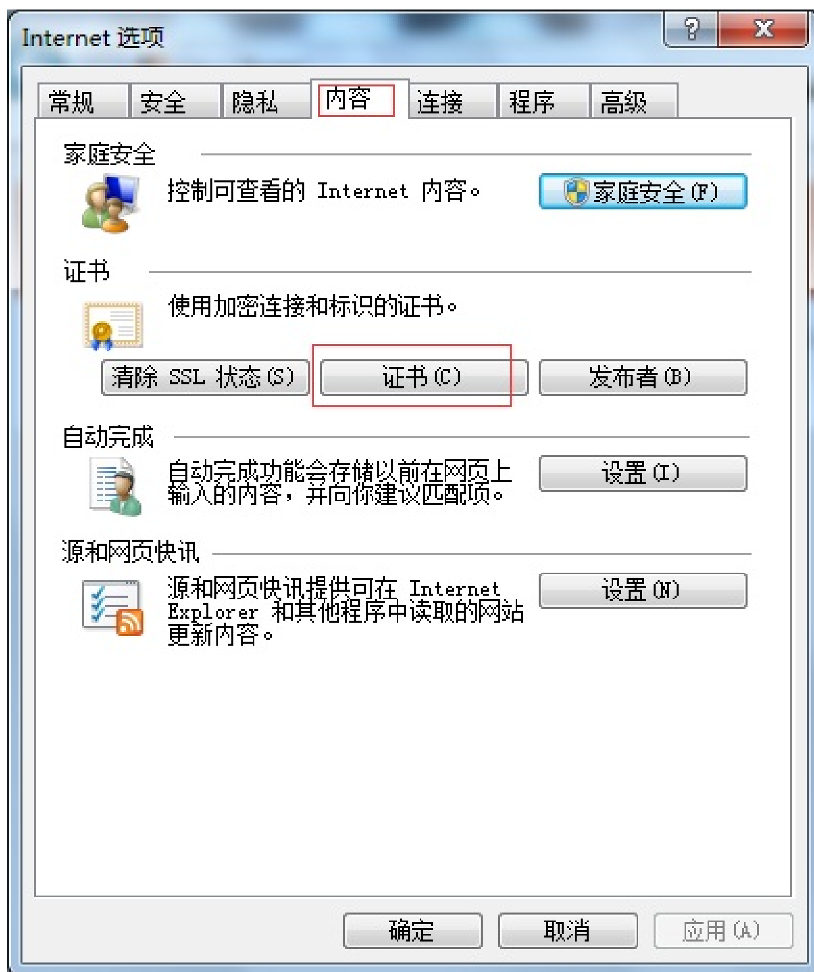
1. 首先，根据前三章内容的学习，你已配置好Burp Proxy监听端口和IE的代理服务器设置。其次，你的IE浏览器中没有安装过Burp Suite的CA证书，如果已经安装，请先卸载证书。详细的卸载方法请参考CA证书的卸载章节。
2. 以管理员身份，启动IE浏览器，在地址栏输入http://burp并回车，进入证书下载页面



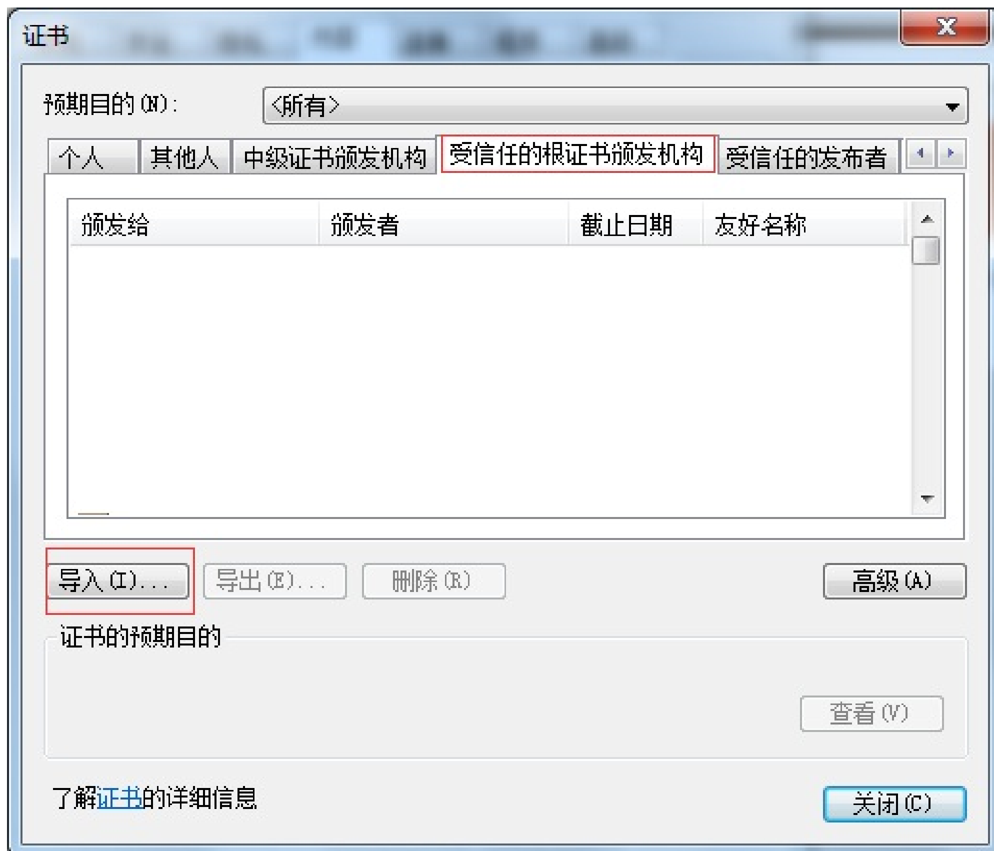
3. 点击上图所示的证书下载，另存为到本地目录。
4. 点击浏览器上的【工具】菜单，打开【Internet选项】。



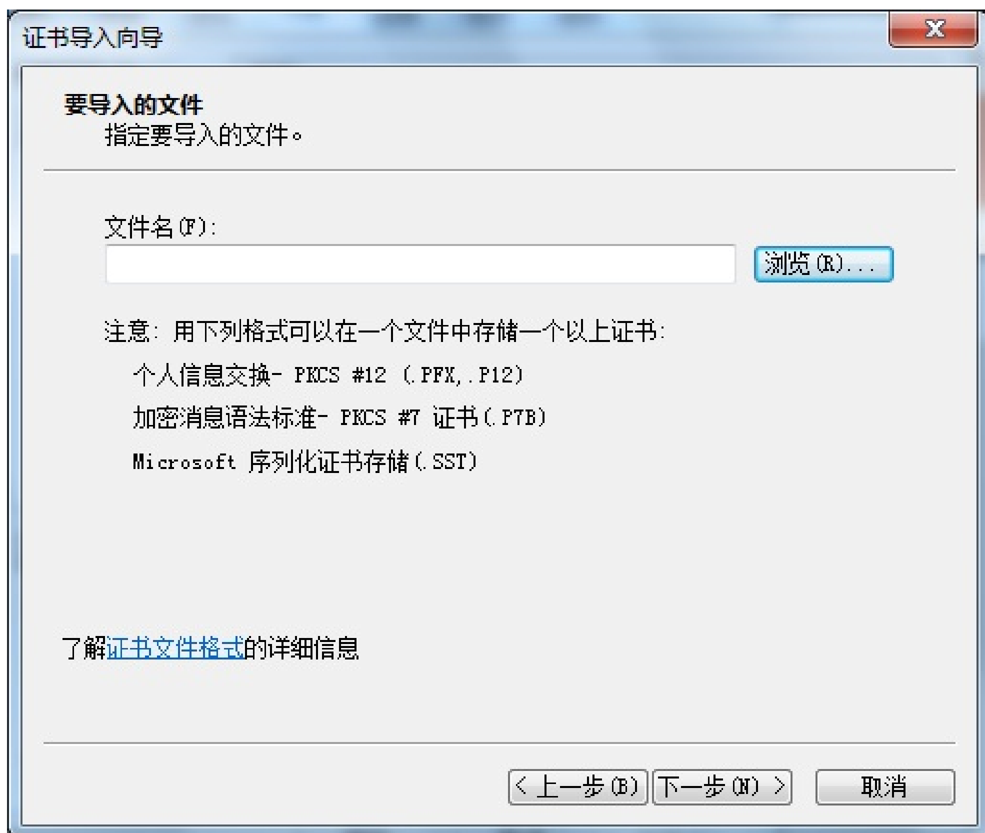
5. 在弹出的证书对话框中，点击【内容】-【证书】。



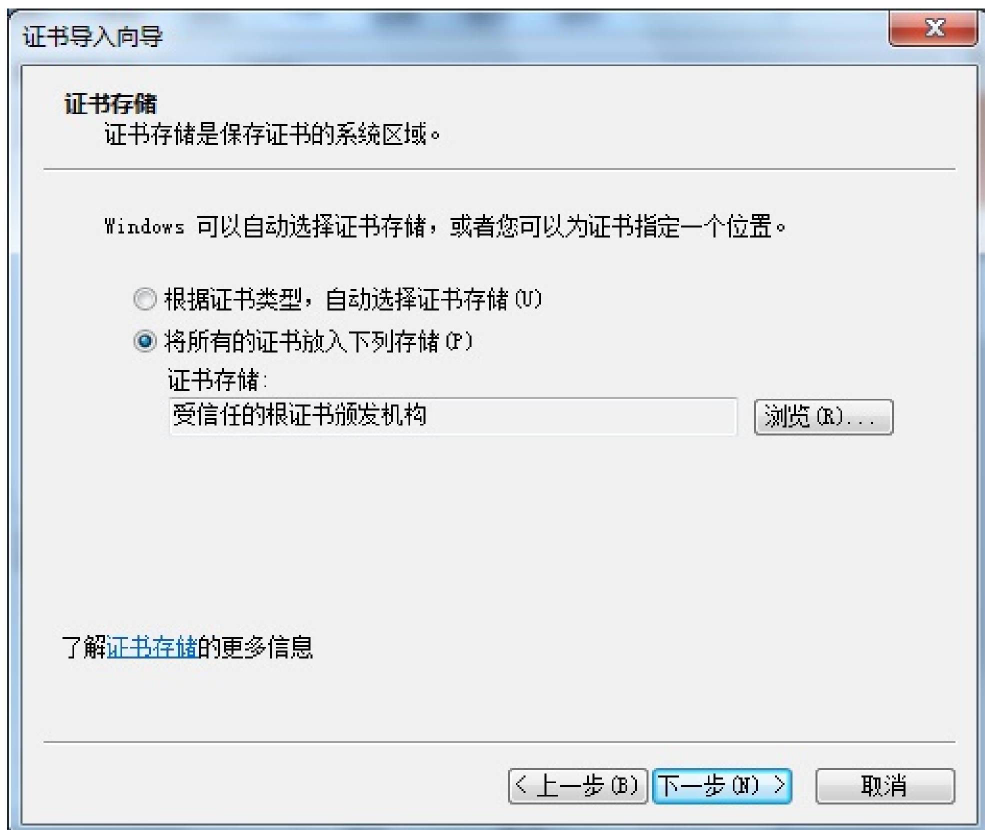
6. 在弹出的证书对话框中，选中【受信任的根证书颁发机构】，点击【导入】。



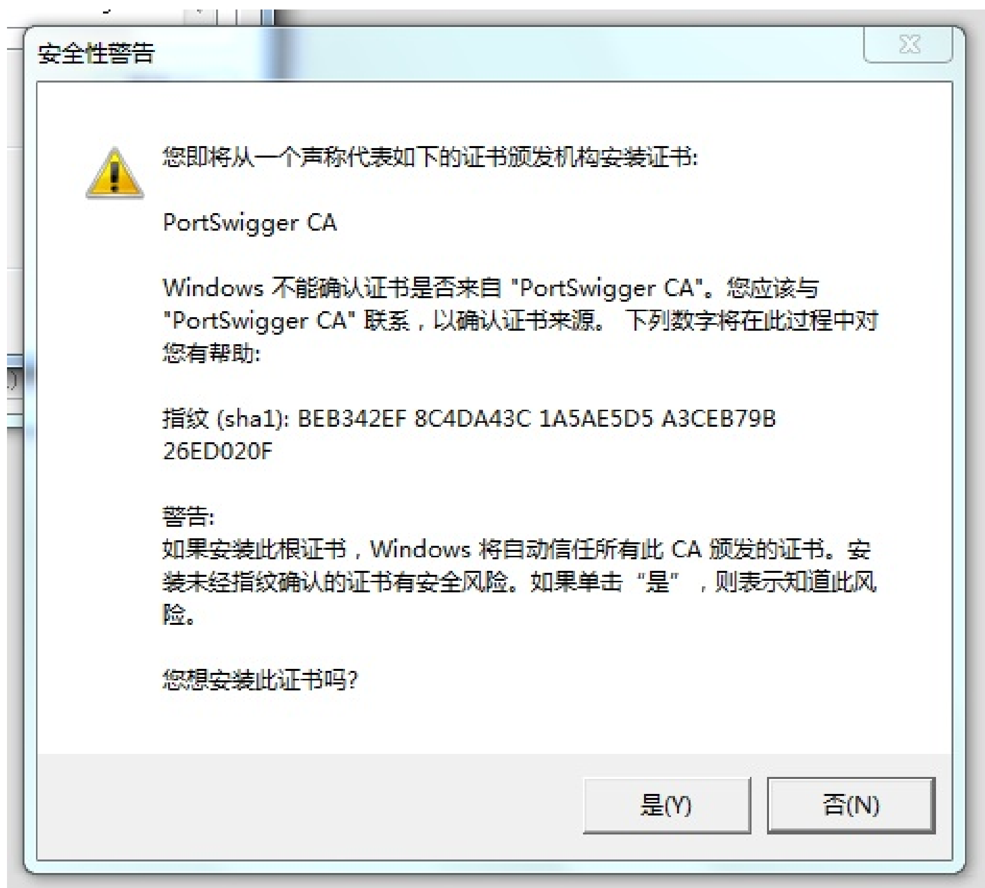
7. 点击【下一步】，选择步骤3保存的证书文件，进行下一步操作。



8. 指定证书的存储位置，如图



9. 点击【下一步】，直至完成。这时，会提示安全警告，点击【是】，提示导入完成。

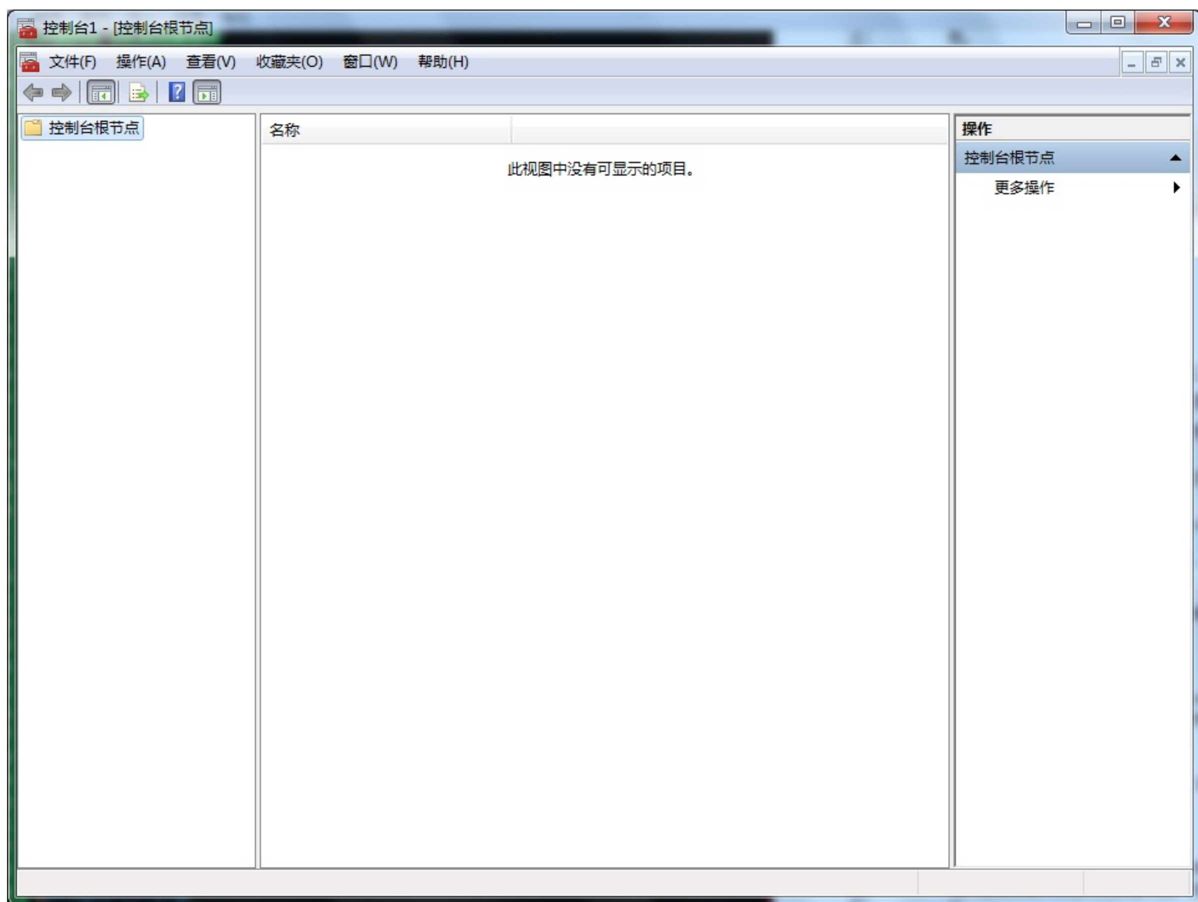


10. 关闭IE，重启浏览器，CA证书即配置完成。

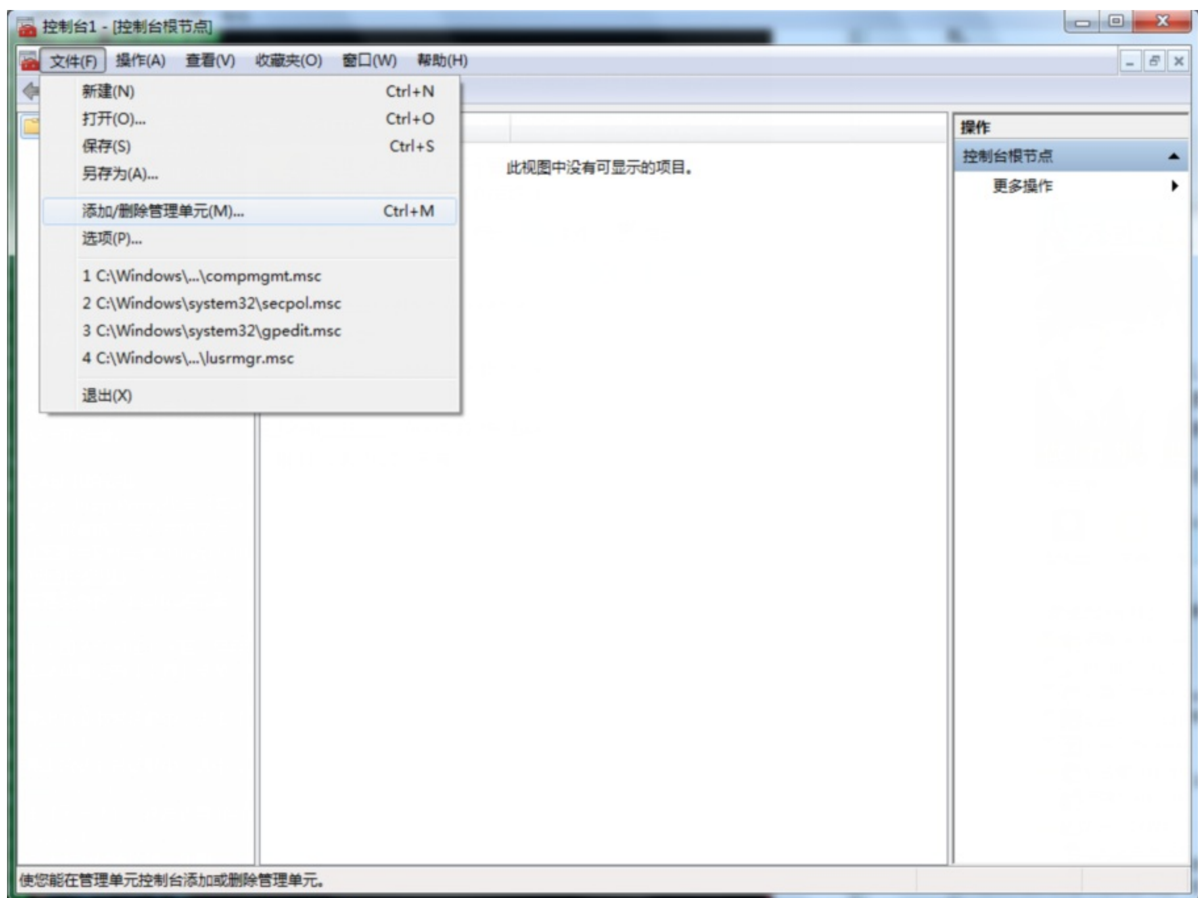
CA证书的卸载

CA证书的卸载的通常有两种方式，第一种方式在上一章节CA证书安装中的第6步，找到需要卸载的证书，点击【删除】即可。我们这里主要描述第二种删除方式，主要是为了解决在第一种方式的基础上删除按钮失效或者证书列表里看不到的证书也一起删除的方法。

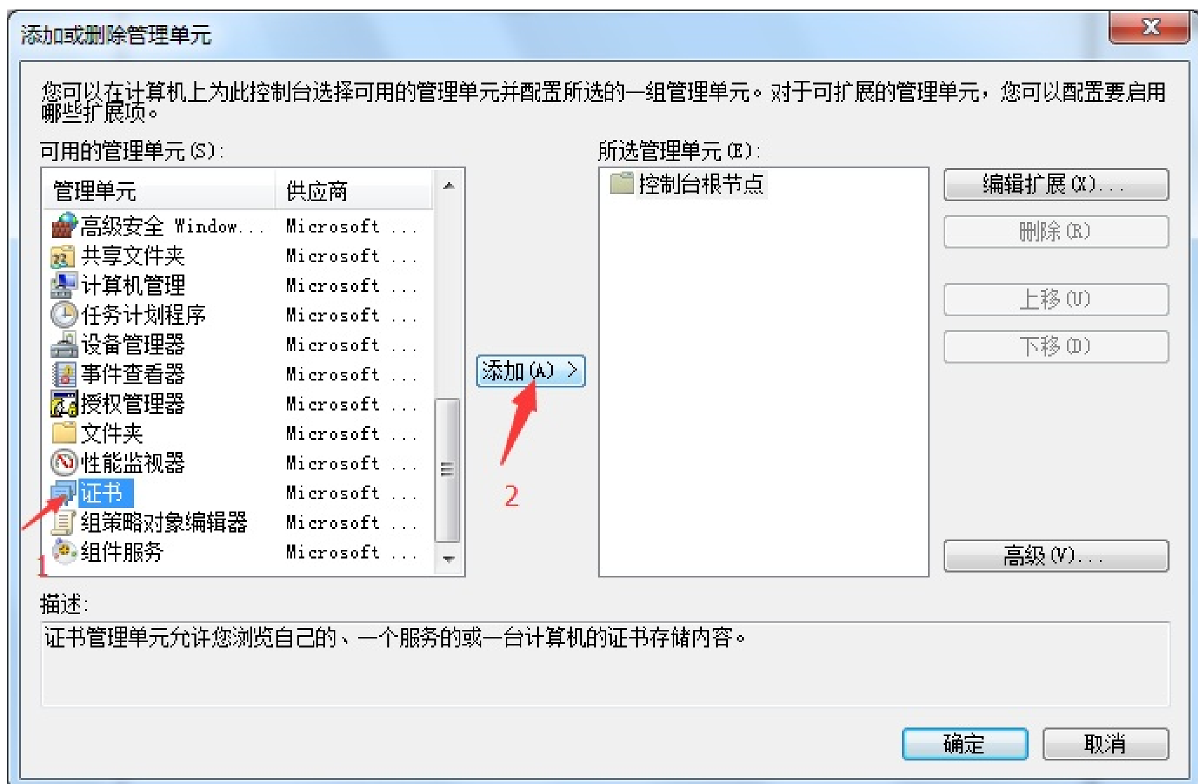
1. 首先，我们打开cmd，输入mmc，或者你在运行输入框里直接输入mmc回车，会弹出管理控制台。



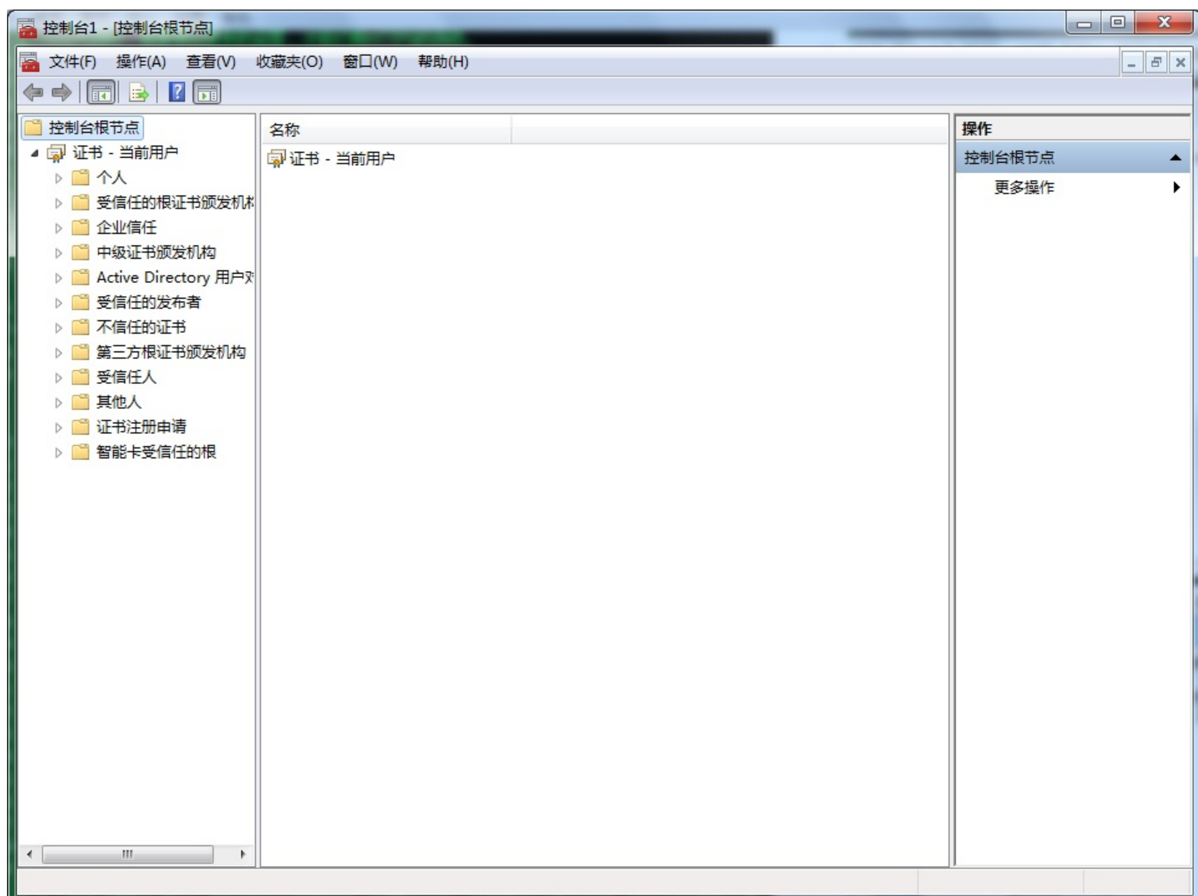
2. 点击【文件】菜单，打开【添加/删除管理单元】



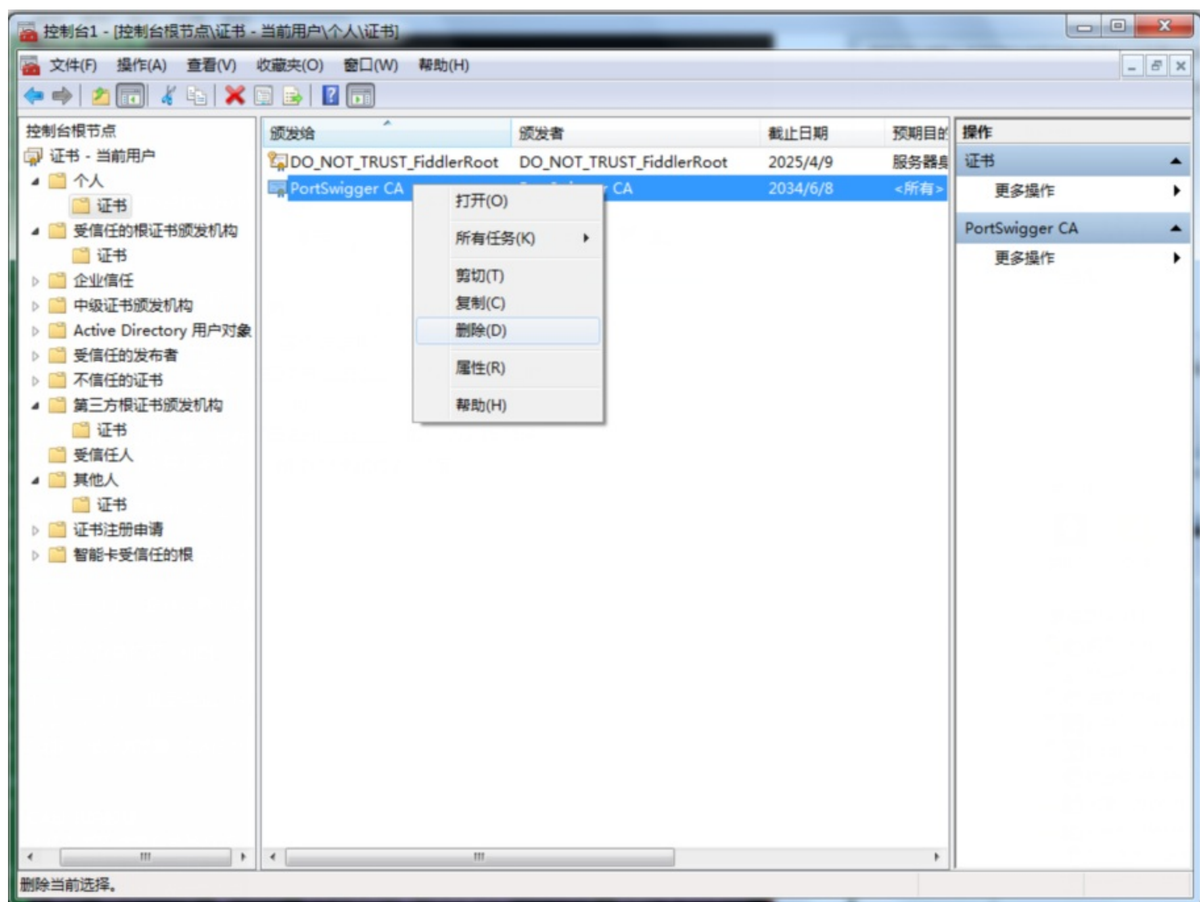
3. 找到证书，如下图1，点击【添加】按钮，如下图2



4. 在弹出的对话框中默认选中【我当前的用户】，点击【完成】，一直到结束，这是会在控制台跟节点下多了一个证书的节点。



5. 打开CA证书所在的位置，选择删除即可。



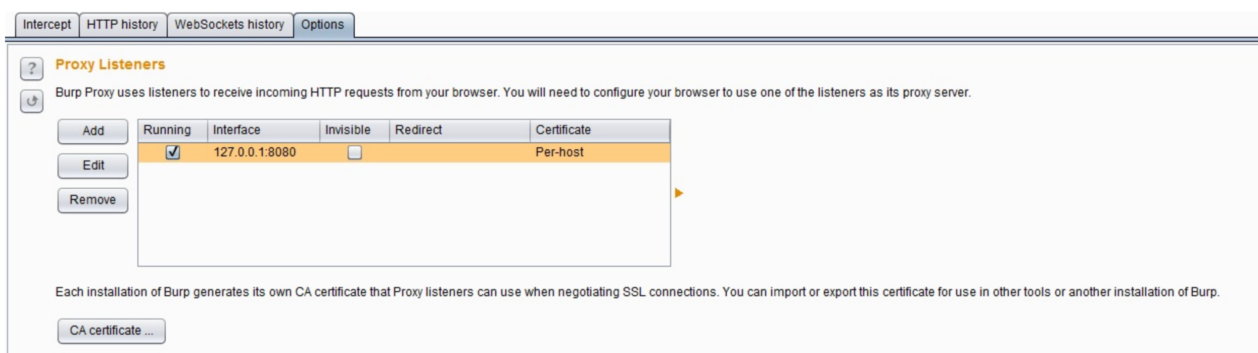
6. 这时，你再返回到IE浏览器的证书列表里，则不会再看到被删除的证书了。

除了IE之外，其他的浏览器如FireFox、Chrome、Sarifa等都证书的安装和卸载基本类似，操作时可以以IE的CA证书安装作为参考。

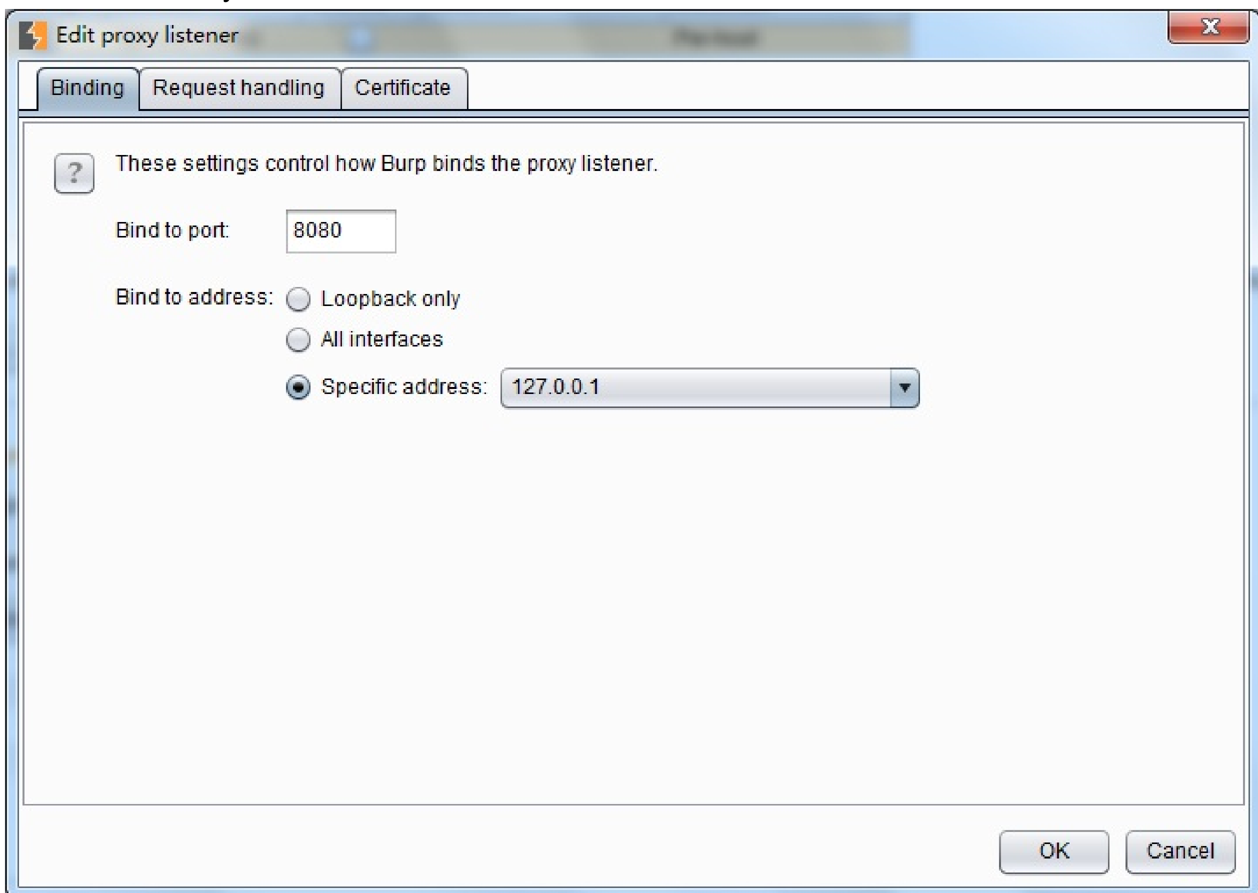
Proxy监听设置

当我们启动Burp Suite时，默认会监听本地回路地址的8080端口，除此之外，我们也可以在默认监听的基础上，根据我们自己的需求，对监听端口和地址等参数进行自由设置。特别是当我们测试非浏览器应用时，无法使用浏览器代理的方式去拦截客户端与服务器端通信的数据流量，这种情况下，我们会使用自己的Proxy监听设置，而不会使用默认设置。

- Proxy监听设置



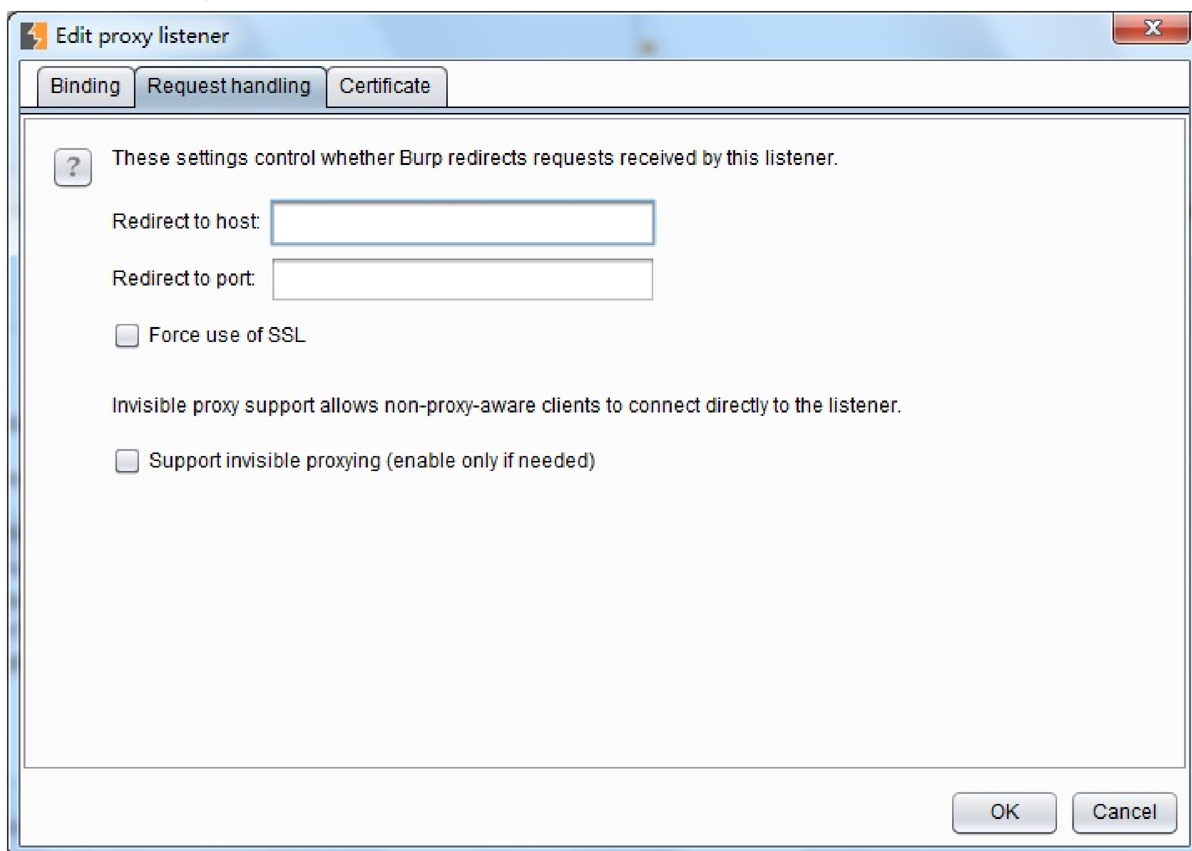
当我们在实际使用中，可能需要同时测试不同的应用程序时，我们可以通过设置不同的代理端口，来区分不同的应用程序，Proxy监听即提供这样的功能设置。点击图中的【Add】按钮，会弹出Proxy监听设置对话框，里面有更丰富的设置，满足我们不同的测试需求。



Proxy监听设置主要包含3块功能：

1. 端口和IP绑定设置Binding 绑定的端口port是指Burp Proxy代理服务监听的端口，绑定IP地址分仅本地回路、所有接口、指定地址三种模式，在渗透测试中，无论你选择哪种模式，你需要明白一点，当你选择的非本地回路IP地址时，同局域网内的其他电脑也可以访问你的监听地址。

2. 请求处理Request Handling 请求处理主要是用来控制接受到Burp Proxy监听端口的请求后，如果对请求进行处理的。



其具体配置可分为：端口的转发、主机名/域名的转发、强制使用SSL和隐形代理4个部分。当我们配置了端口的转发时，所有的请求都会被转发到这个端口上；如果我们配置了主机或域名的转发，则所有的请求会转发到指定的主机或域名上。同时，我们可以指定，通过Burp Proxy的消息是否强制使用SSL，如果设置了此项，则请求若是http协议，经Burp proxy代理后将转换为https协议。隐形代理主要是用于测试富客户端应用或者是非浏览器代理方式的应用，当我们设置了它,访问这些应用时，将通过非代理的方式，直接连接Burp Proxy的监听端口。

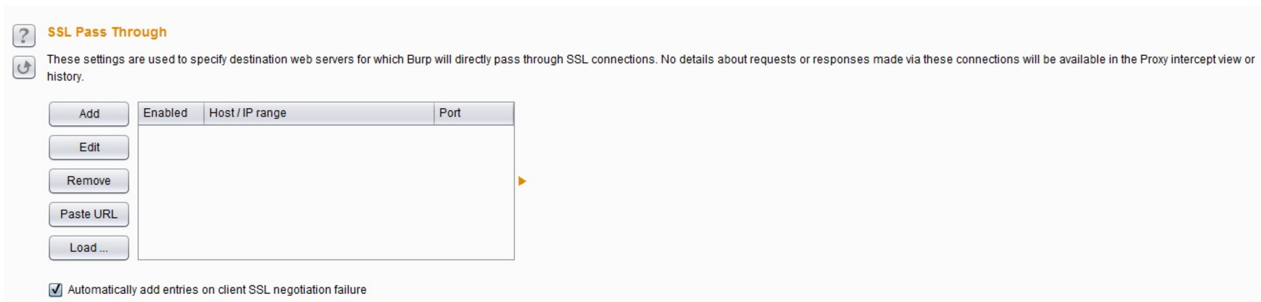
3. SSL 证书 这些设置控制呈现给SSL客户端的服务器SSL证书。可以解决使用拦截代理时出现的一些SSL问题： 1.您可以消除您的浏览器的SSL警报，并需要建立SSL例外。其中，网页加载来自其他域的SSL保护的项目，可以确保这些正确的加载到浏览器，而不需要为每个域手动接受代理的SSL证书。 2.可以与该拒绝无效的SSL证书连接到服务器胖客户机应用程序的工作。 它有下列选项可供设置：

4. 使用自签名证书（Use a self-signed certificate ） —— 一个简单的自签名SSL证书呈现给您的浏览器，它总是会导致SSL警告。
5. 生成每个主机的CA签名证书（Generate CA-signed per-host certificates）—— 这是默认选项。在安装时，Burp创造了一个独特的自签名的证书颁发机构（CA）证书，并将此计算机上使用。当你的浏览器发出的SSL连接指定主机，Burp生成该主机的SSL证书，由CA证书签名。您可以安装Burp的CA证书作为浏览器中的受信任的根，从而使每个主机证书没有任何警报接受。

6. 生成与特定的主机名CA签发的证书（Generate a CA-signed certificate with a specific hostname）—— -是类似于前面的选项;不同的是，Burp会生成一个主机证书与每一个SSL连接使用，使用指定的主机名。
7. 使用自定义证书（Use a custom certificate）—— 此选项可以加载一个特定的证书（在PKCS＃12格式）呈现给浏览器。如果应用程序使用这需要一个特定的服务器证书（例如，与给定的序列号或证书链）的客户端应该使用这个选项。

SSL直连和隐形代理

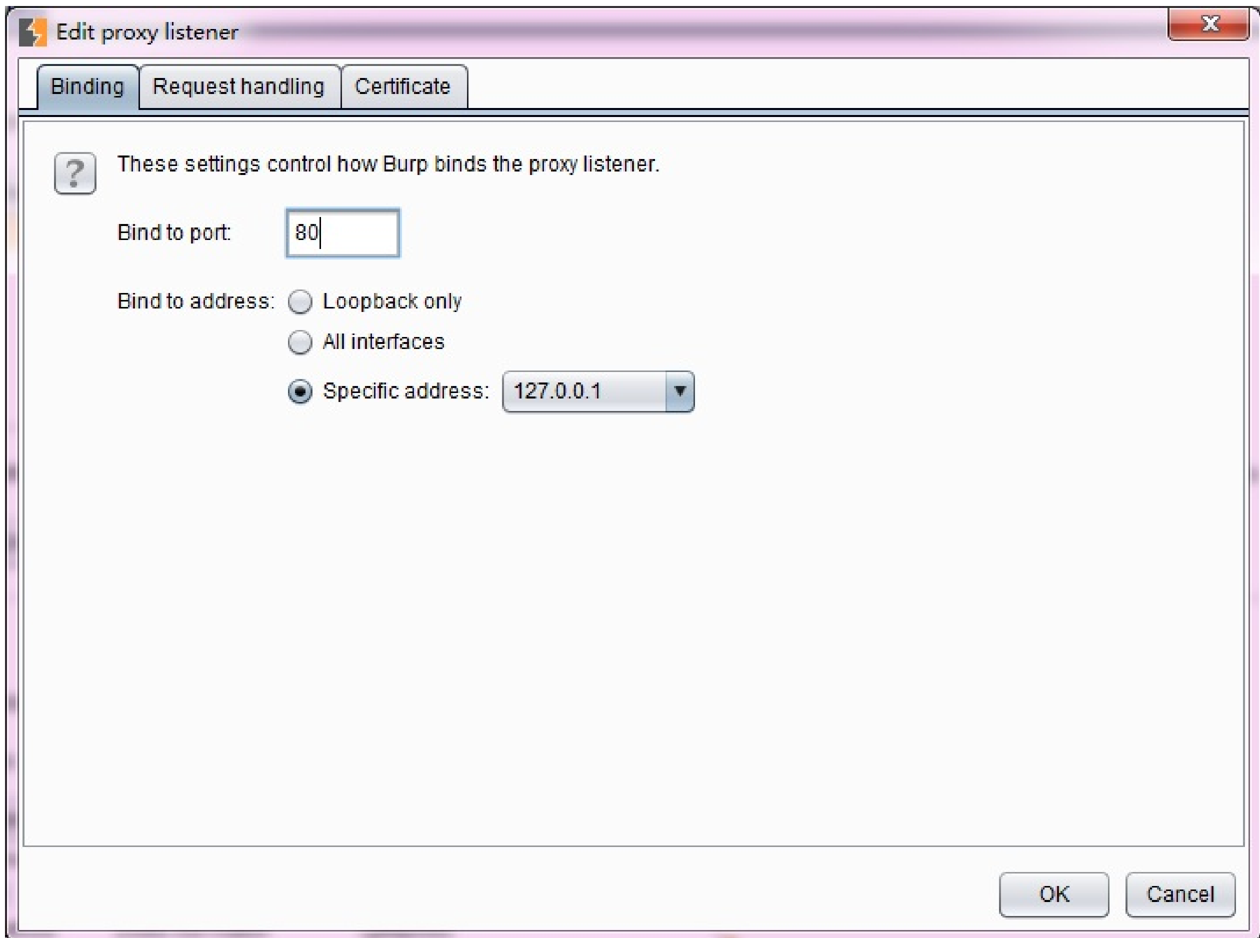
SSL直连的设置主要用于指定的目的服务器直接通过SSL连接，而通过这些连接的请求或响应任何细节将在Burp代理拦截视图或历史日志中可见。通过SSL连接传递可以并不是简单地消除在客户机上SSL错误的情况下有用。比如说，在执行SSL证书的移动应用。如果应用程序访问多个域，或使用HTTP和HTTPS连接的混合，然后通过SSL连接到特定的主机问题仍然使您能够以正常的方式使用Burp的其他方式进行通信。如果启用自动添加客户端SSL协商失败的选项，当客户端失败的SSL协议检测（例如，由于不承认Burp的CA证书），并会自动将相关的服务器添加到SSL直通通过列表中去。其设置界面如下图所示：



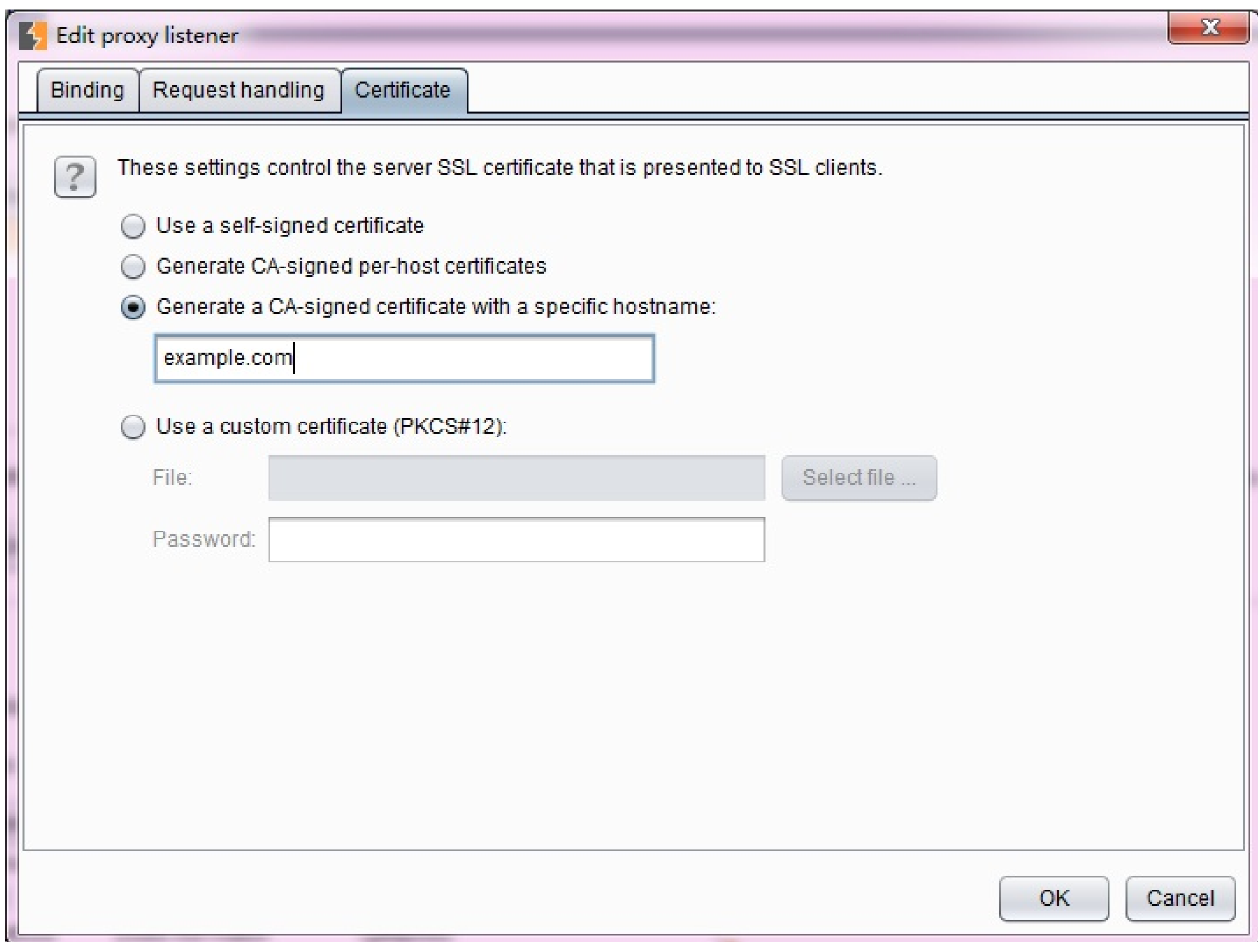
有时候，在拦截富客户端软件时，我们通常需要使用隐形代理。富客户端软件通常是指运行在浏览器之外的客户端软件，这就意味着它本身不具有HTTP代理是属性。当它进行网络通信时，客户端将无法使代理感知或者无法由代理进行通信。在Burp中，我们可以使用隐形代理的方式，对通信内容进行代理或拦截，从而对通信的请求和响应消息进行分析。使用隐形代理通常需要做如下设置（以https://example.com为例）： 1.配置hosts文件，Windows操作系统下的目录位置Windows/System32/drivers/etc/hosts，而Linux或者Unix下的目录为/etc/hosts，添加如下行：

```
127.0.0.1 example.com
```

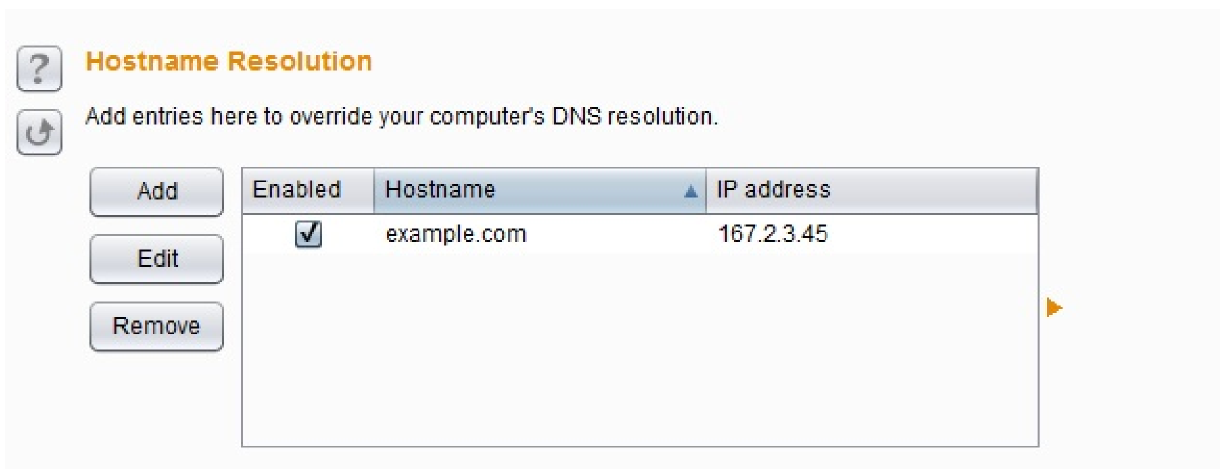
2.第一步设置完成之后，我们需要添加一个新的监听来运行在HTTP默认的80端口，如果通信流量使用HTTPS协议，则端口为443。



3.如果是HTTPS协议的通信方式，我们需要一个指定域名的CA证书。



4.接着，我们需要把Burp拦截的流量转发给原始请求的服务器。这需要在Options->Connections->Hostname Resolution 进行设置。因为我们已经告诉了操作系统，example.com的监听地址在127.0.0.1上，所以我们必须告诉Burp，将example.com的流量转发到真实的服务器那里去。



5.

通过这样的配置，我们就可以欺骗富客户端软件，将流量发送到Burp监听的端口上，再由Burp将流量转发给真实的服务器。

第五章 如何使用Burp Target

Burp Target 组件主要包含站点地图、目标域、Target 工具三部分组成，他们帮助渗透测试人员更好地了解目标应用的整体状况、当前的工作涉及哪些目标域、分析可能存在的攻击面等信息，下面我们就分别来看看Burp Target的三个组成部分。

本章的主要内容有：

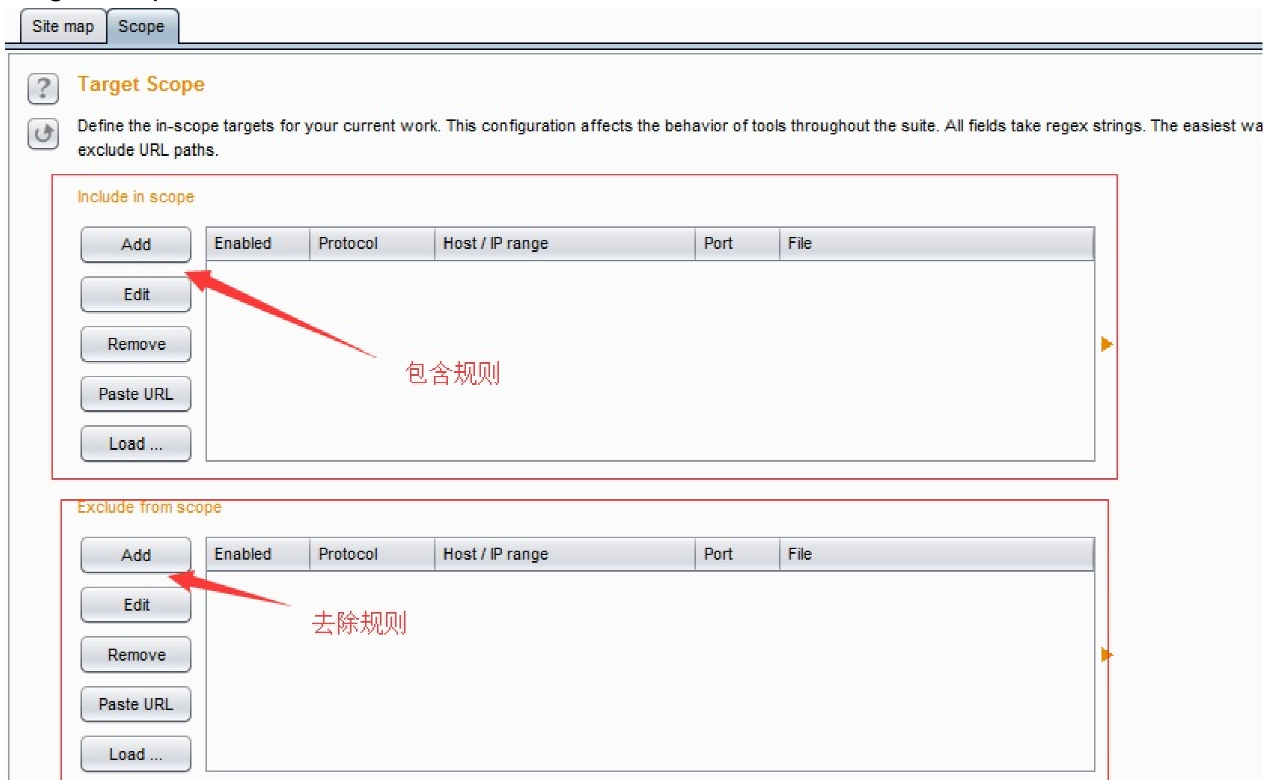
- 目标域设置 Target Scope
- 站点地图 Site Map
- Target 工具的使用

目标域设置 Target Scope

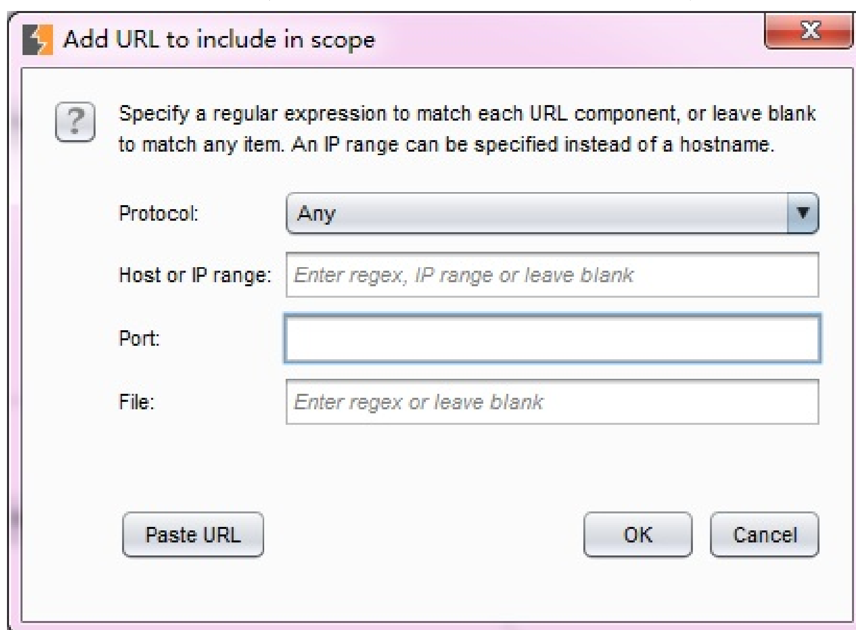
Target Scope中作用域的定义比较宽泛，通常来说，当我们对某个产品进行渗透测试时，可以通过域名或者主机名去限制拦截内容，这里域名或主机名就是我们说的作用域；如果我们想限制得更为细粒度化，比如，你只想拦截login目录下的所有请求，这时我们也可以在此设置，此时，作用域就是目录。总体来说，Target Scope主要使用于下面几种场景中：

- 限制站点地图和Proxy 历史中的显示结果
- 告诉Burp Proxy 拦截哪些请求
- Burp Spider抓取哪些内容
- Burp Scanner自动扫描哪些作用域的安全漏洞
- 在Burp Intruder和Burp Repeater 中指定URL

通过Target Scope 我们能方便地控制Burp 的拦截范围、操作对象，减少无效的噪音。在Target Scope的设置中，主要包含两部分功能：允许规则和去除规则。



其中允许规则顾名思义，即包含在此规则列表中的，视为操作允许、有效。如果此规则用于拦截，则请求消息匹配包含规则列表中的将会被拦截；反之，请求消息匹配去除列表中的将

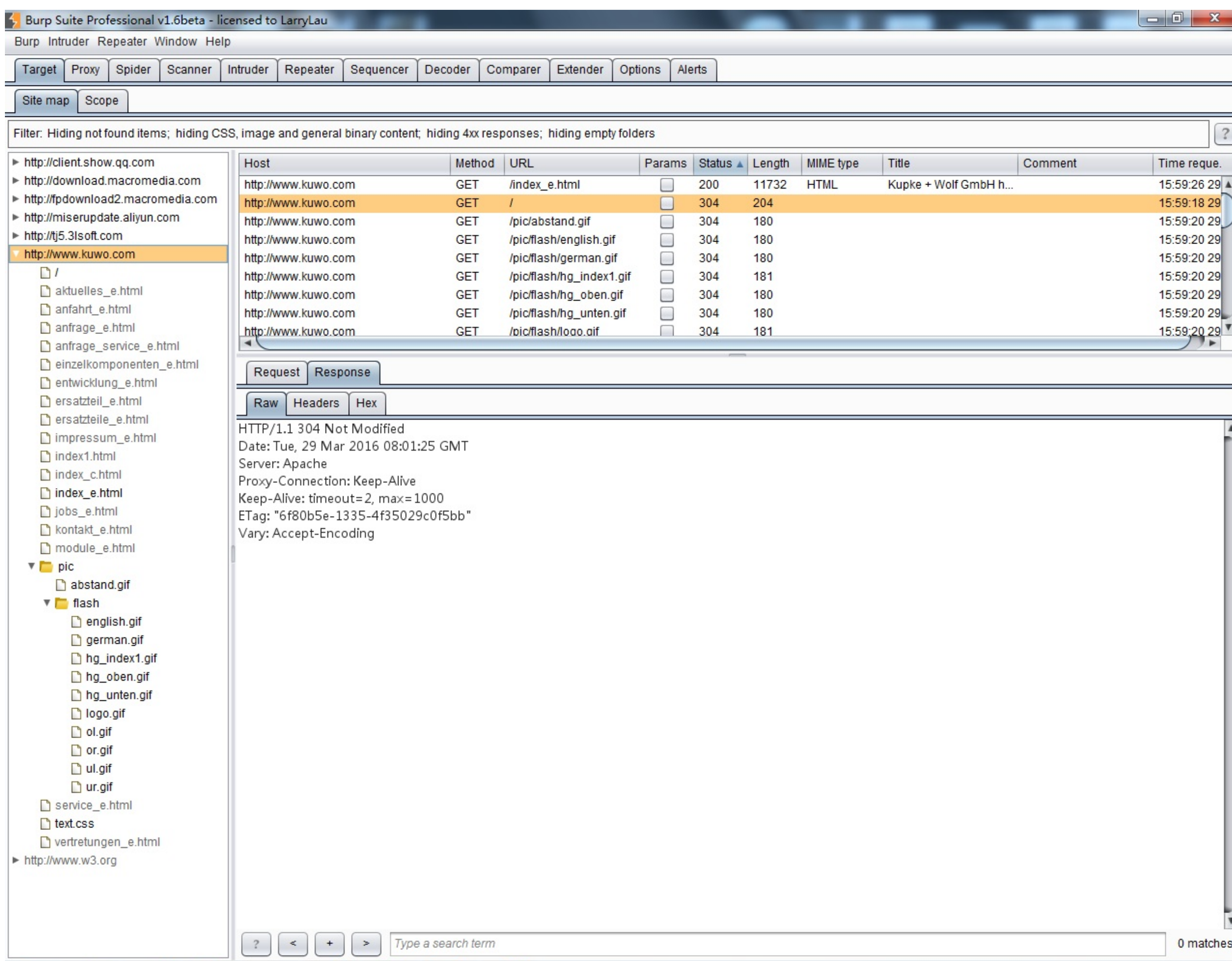


不会被拦截。从上图的添加规则对话框中我们可以看出，规则主要由协议、域名或IP地址、端口、文件名4个部分组成，这就意味着我们可以从协议、域名或IP地址、端口、文件名4个维度去控制哪些消息出现在允许或去除在规则列表中。

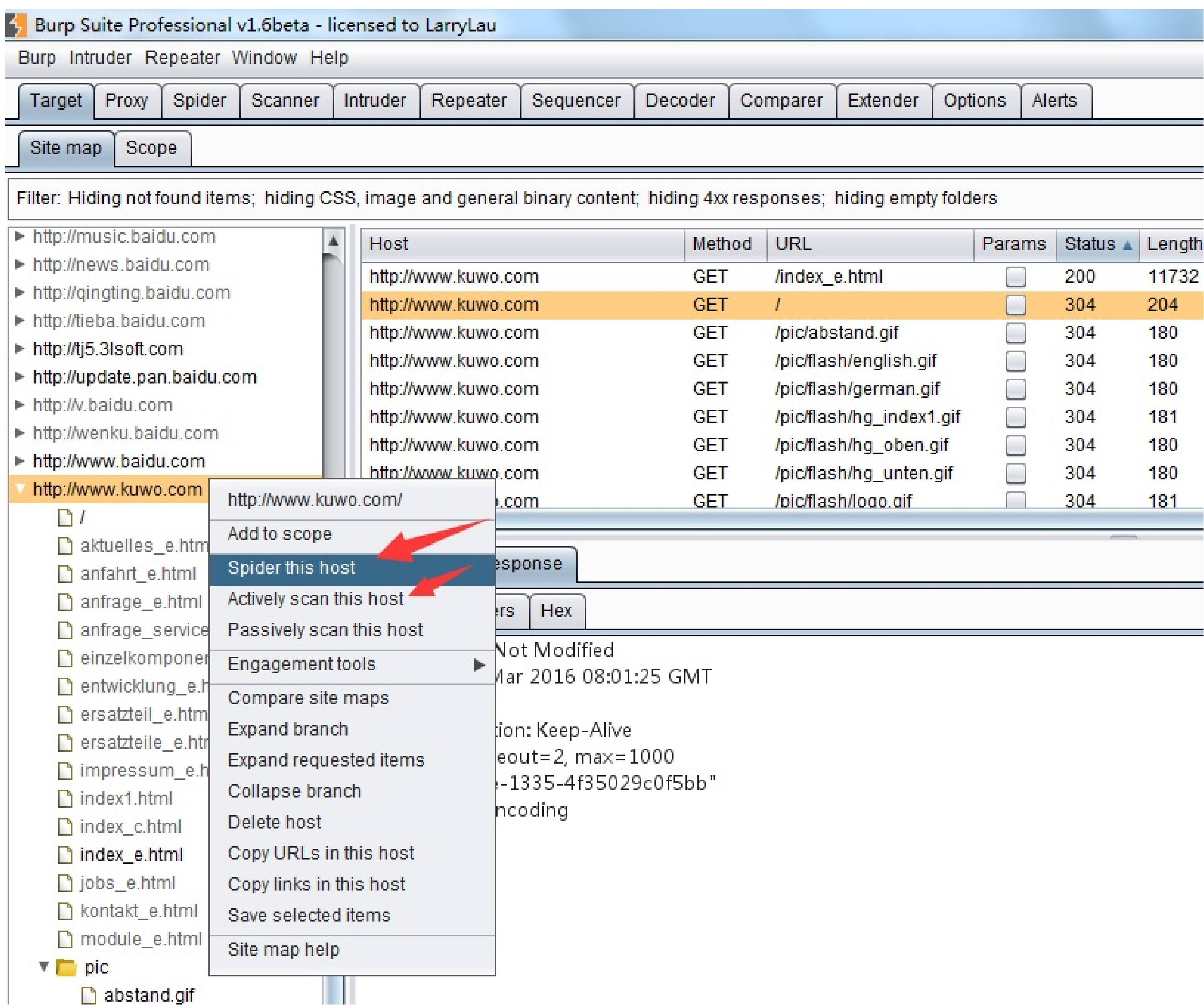
当我们设置了Target Scope（默认全部为允许），使用Burp Proxy进行代理拦截，在渗透测试中通过浏览器代理浏览应用时，Burp会自动将浏览信息记录下来，包含每一个请求和应答的详细信息，保存在Target站点地图中。

站点地图 Site Map

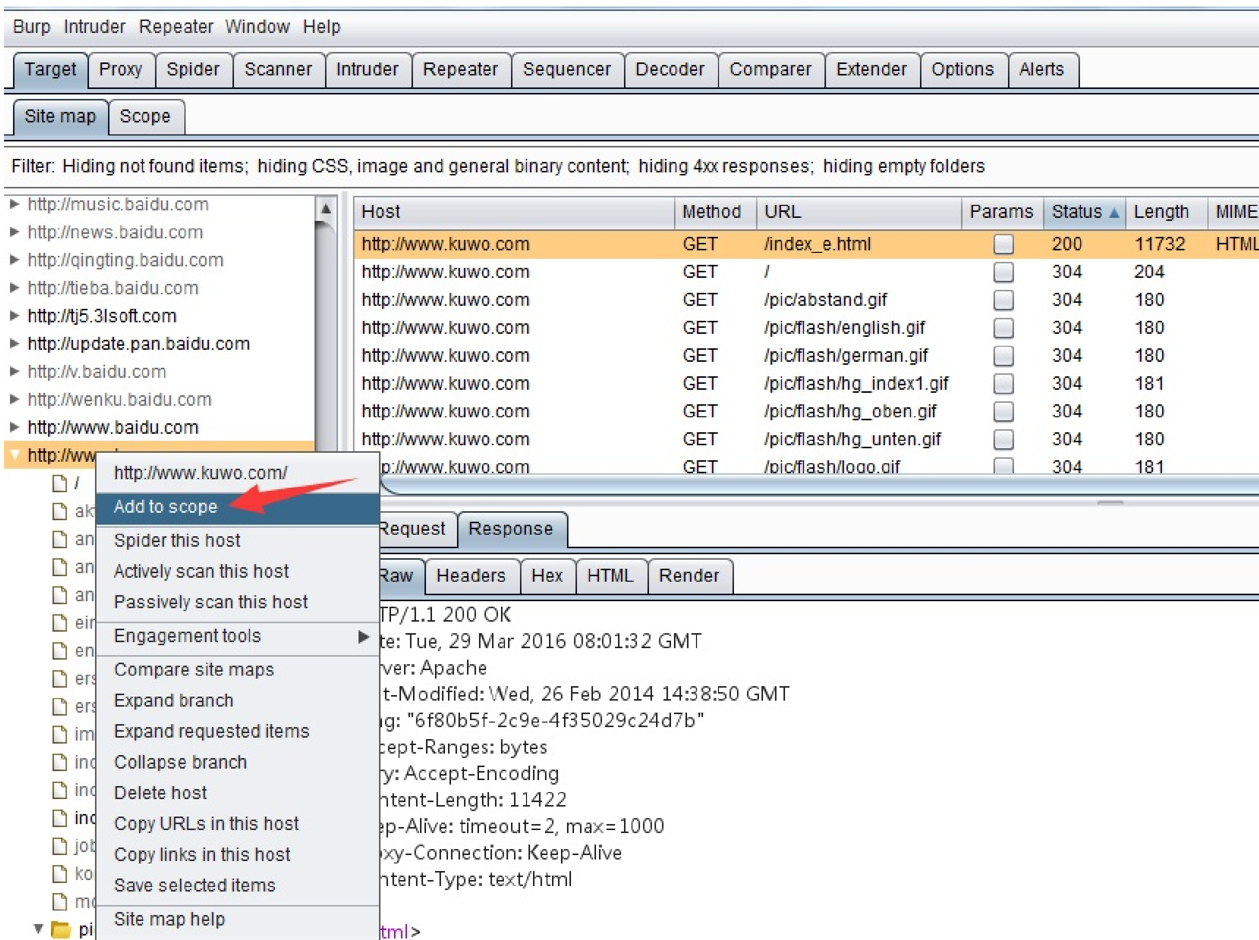
下图所示站点地图为一次渗透测试中，通过浏览器浏览的历史记录在站点地图中的展现结果。



从图中我们可以看出，Site Map的左边为访问的URL，按照网站的层级和深度，树形展示整个应用系统的结构和关联其他域的url情况；右边显示的是某一个url被访问的明细列表，共访问哪些url，请求和应答内容分别是什么，都有着详实的记录。 基于左边的树形结构，我们可以选择某个分支，对指定的路径进行扫描和抓取。



同时，我们也可以将某个域直接加入 Target Scope中.



除了加入 Target Scope外，从上图中，我们也可以看到，对于站点地图的分层，可以通过折叠和展开操作，更好的分析站点结构。

Target 工具的使用

Target 工具的使用的使用主要包括以下部分：

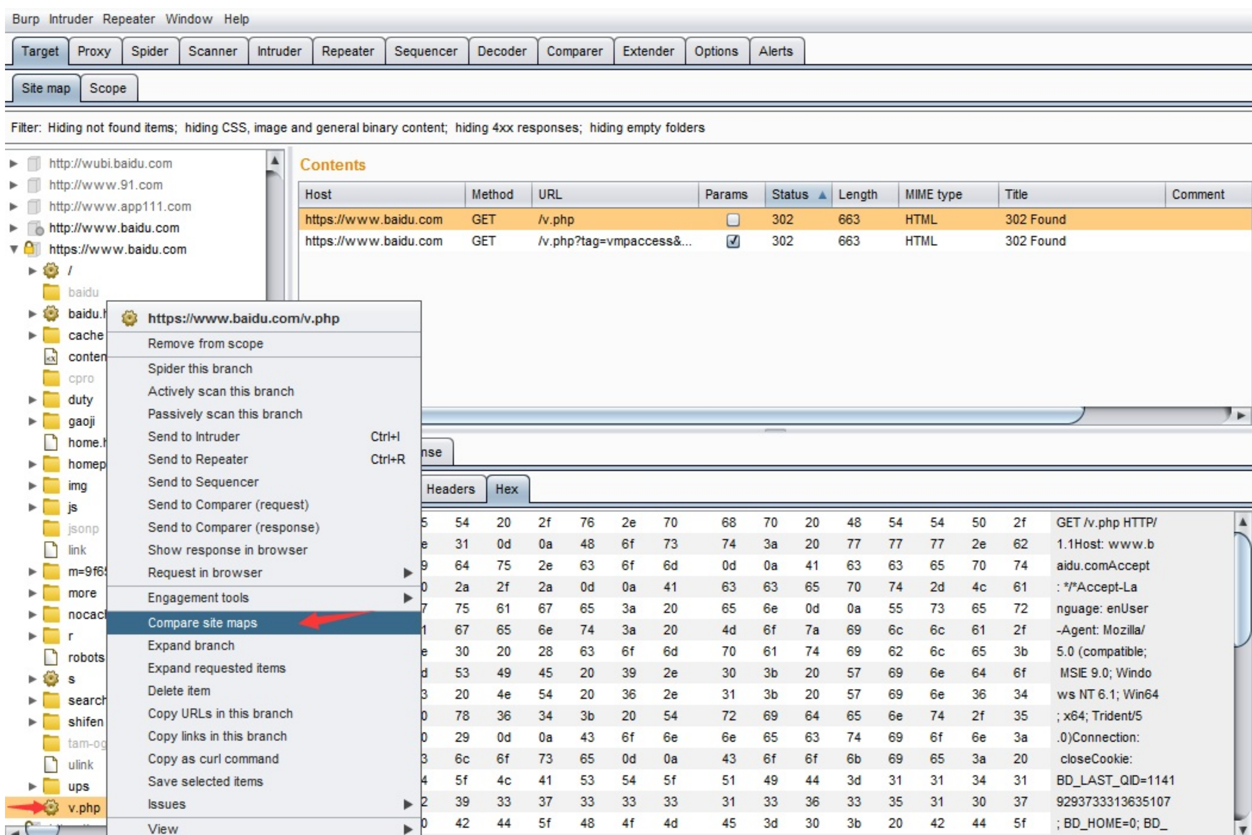
- 手工获取站点地图
- 站点比较
- 攻击面分析

当我们手工获取站点地图时，需要遵循以下操作步骤： 1.设置浏览器代理和Burp Proxy代理，并使之能正常工作。 2.关闭Burp Proxy的拦截功能。 3.手工浏览网页，这时，Target会自动记录站点地图信息。 手工获取站点地图的方式有一个好处就是，我们可以根据自己的需要和分析，自主地控制访问内容，记录的信息比较准确。与自动抓取相比，则需要更长的时间，如果需要渗透测试的产品系统是大型的系统，则对于系统的功能点依次操作一遍所需要的精力和时间对渗透测试人员来说付出都是很大的。

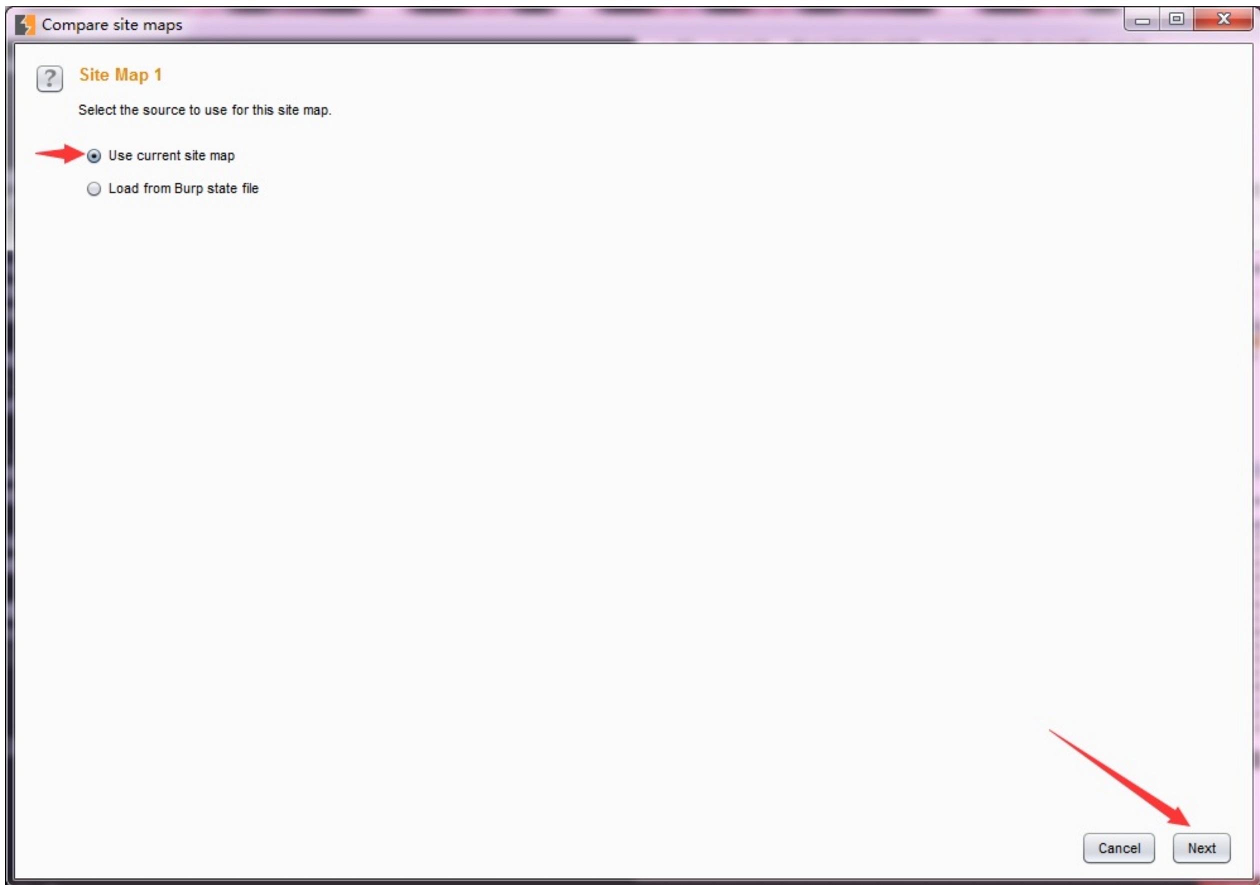
站点比较是一个Burp提供给渗透测试人员对站点进行动态分析的利器，我们在比较帐号权限时经常使用到它。当我们登陆应用系统，使用不同的帐号，帐号本身在应用系统中被赋予了不同的权限，那么帐号所能访问的功能模块、内容、参数等都是不尽相同的，此时使用站点

比较，能很好的帮助渗透测试人员区分出来。一般来说，主要有以下3种场景： 1.同一个帐号，具有不同的权限，比较两次请求结果的差异。 2.两个不同的帐号，具有不同的权限，比较两次请求结果的差异。 3.两个不同的帐号，具有相同的权限，比较两次请求结果的差异。

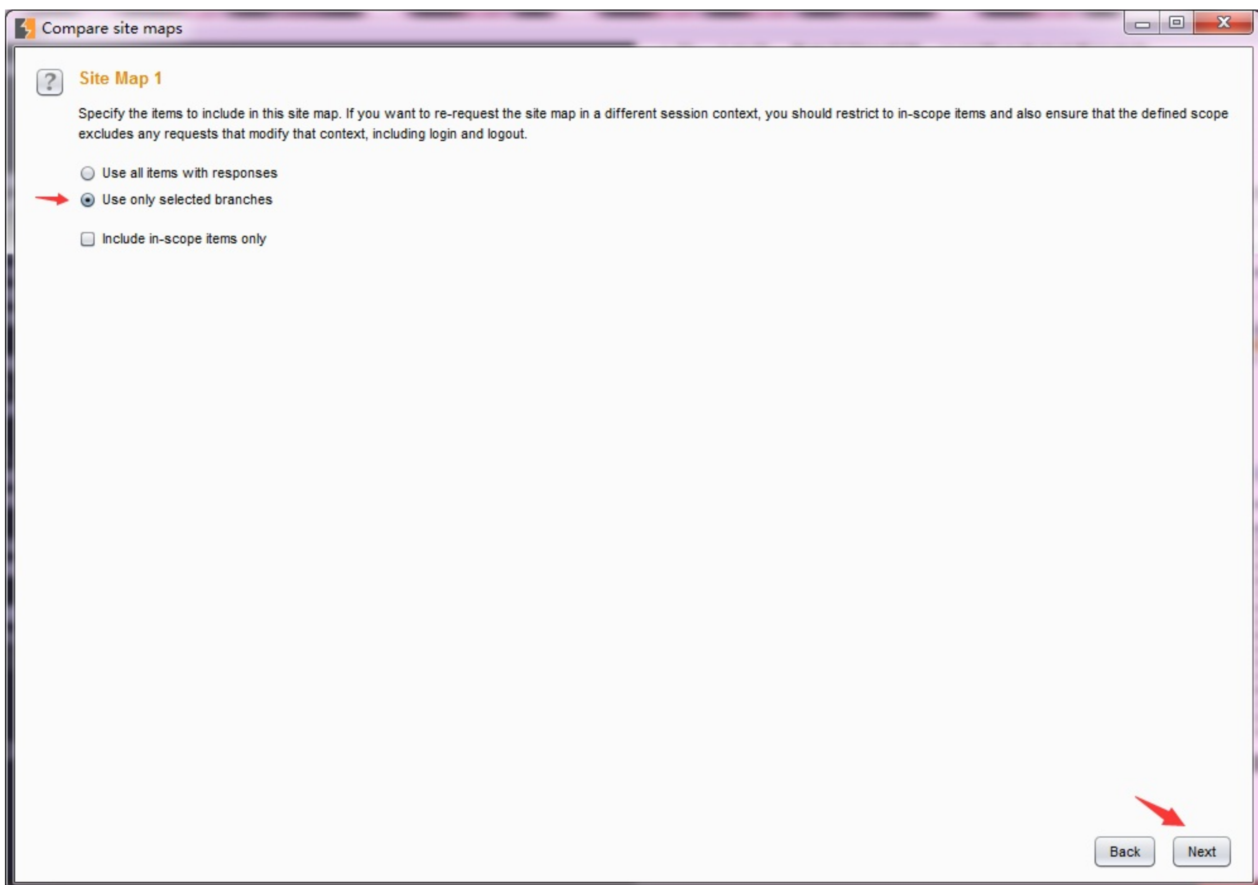
下面我们就一起来看看如何进行站点比较。 1.首先我们在需要进行比较的功能链接上右击，找到站点比较的菜单，点击菜单进入下一步。



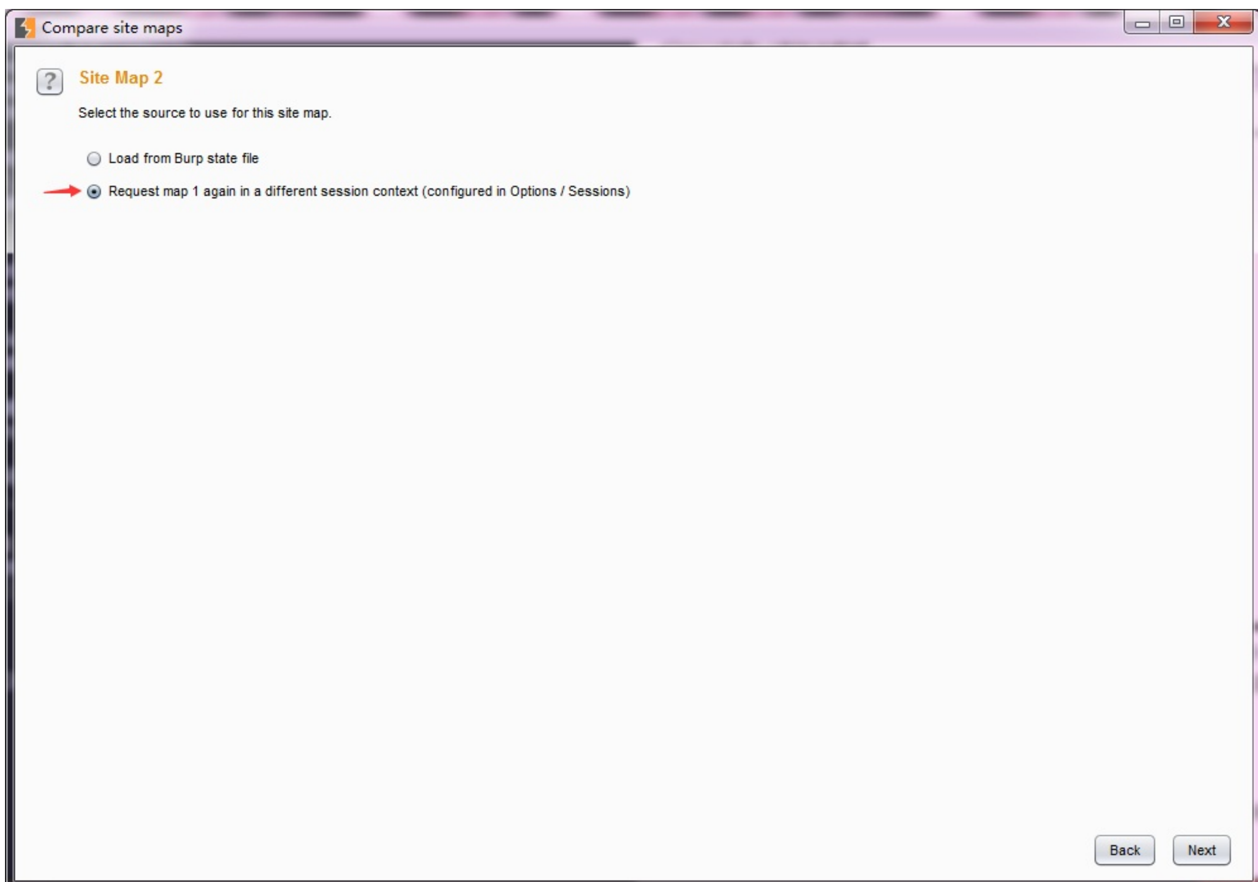
2.由于站点比较是在两个站点地图之间进行的，所以我们在配置过程中需要分别指定Site Map 1和Site Map2。通常情况下，Site Map 1 我们默认为当前会话。如图所示，点击【Next】。



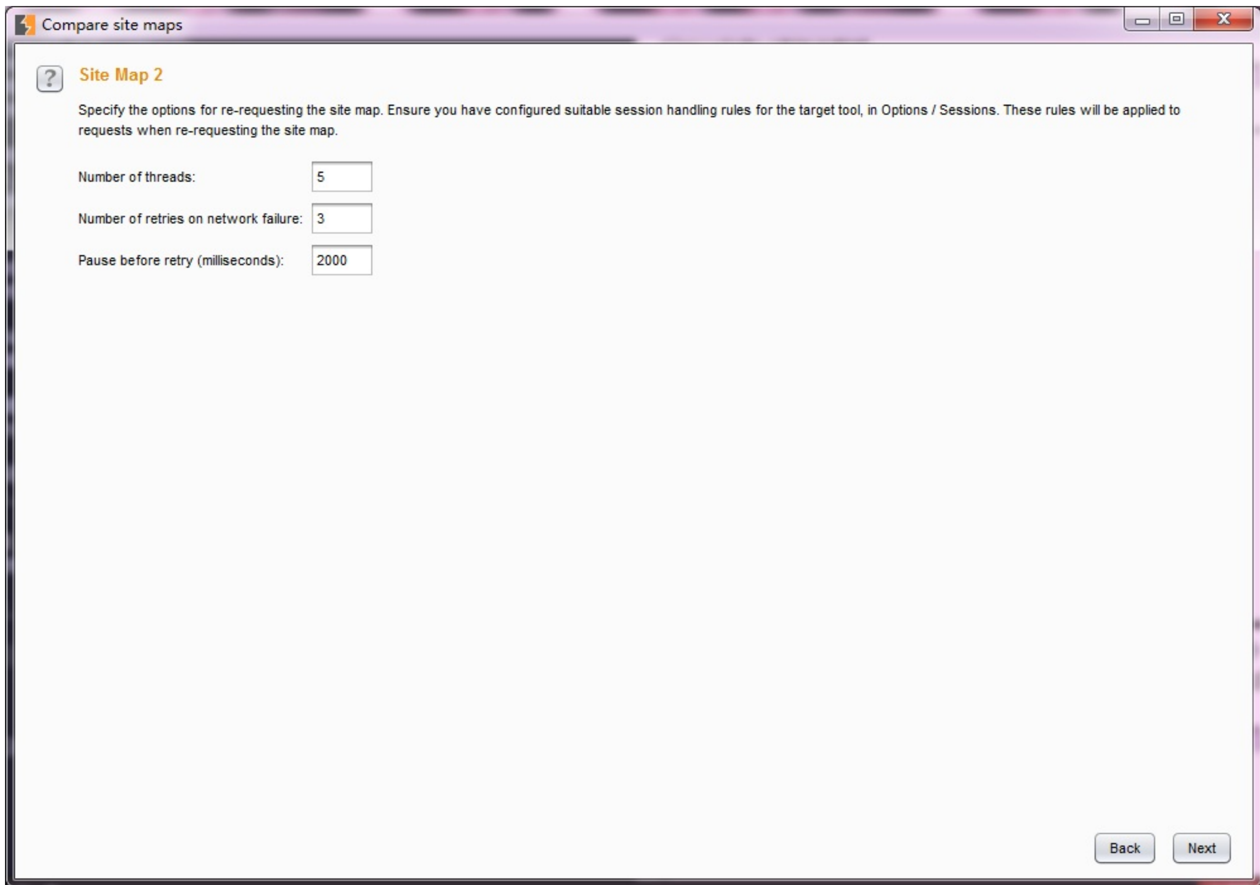
3.这时我们会进入Site Map 1 设置页面，如果是全站点比较我们选择第一项，如果仅仅比较我们选中的功能，则选择第二项。如下图，点击【Next】。如果全站点比较，且不想加载其他域时，我们可以勾选只选择当前域。



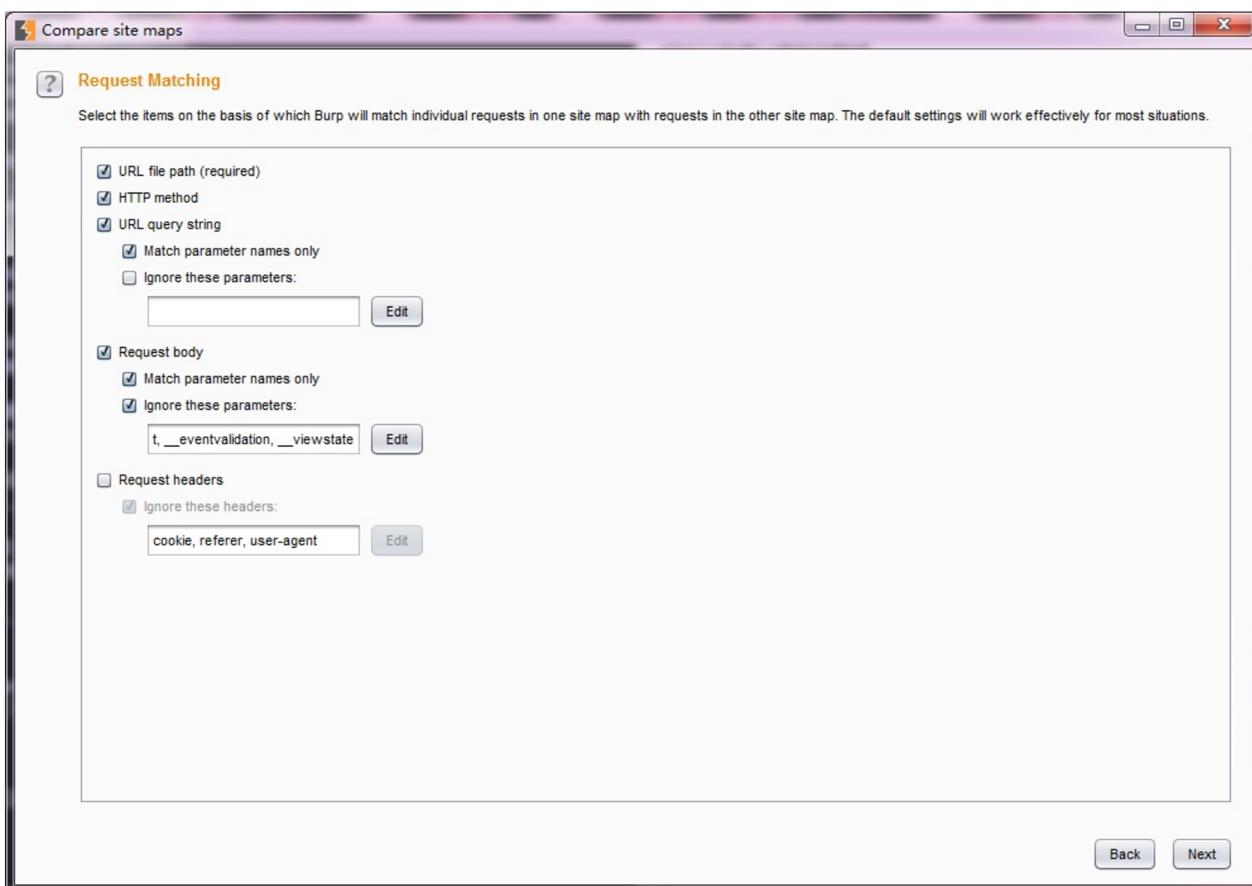
4.接下来就是Site Map 2 的配置，对于Site Map 2我们同样有两种方式，第一种是之前我们已经保存下来的Burp Suite 站点记录，第二种是重新发生一次请求作为Site Map2.这里，我们选择第二种方式。



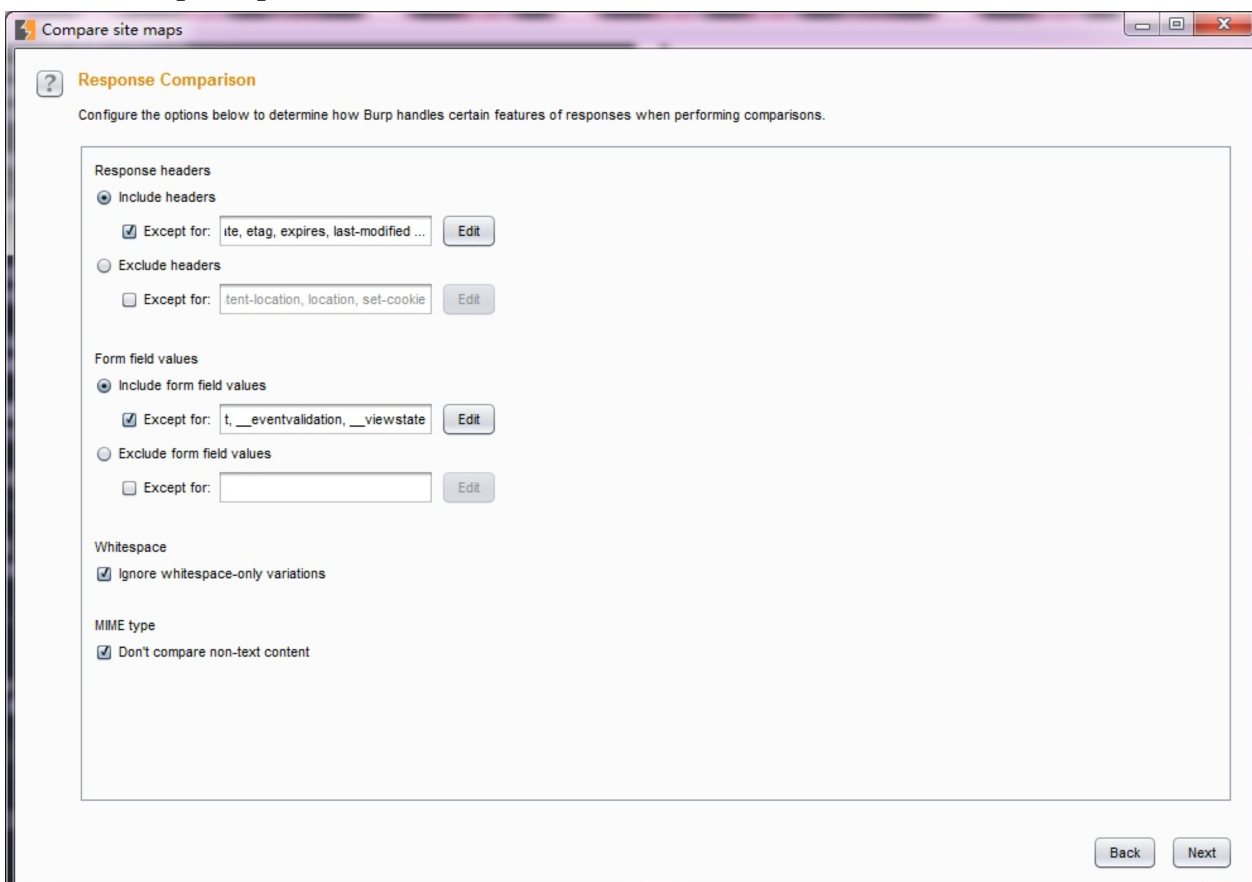
5.如果上一步选择了第二种方式，则进入请求消息设置界面。在这个界面，我们需要指定通信的并发线程数、失败重试次数、暂停的间隙时间。



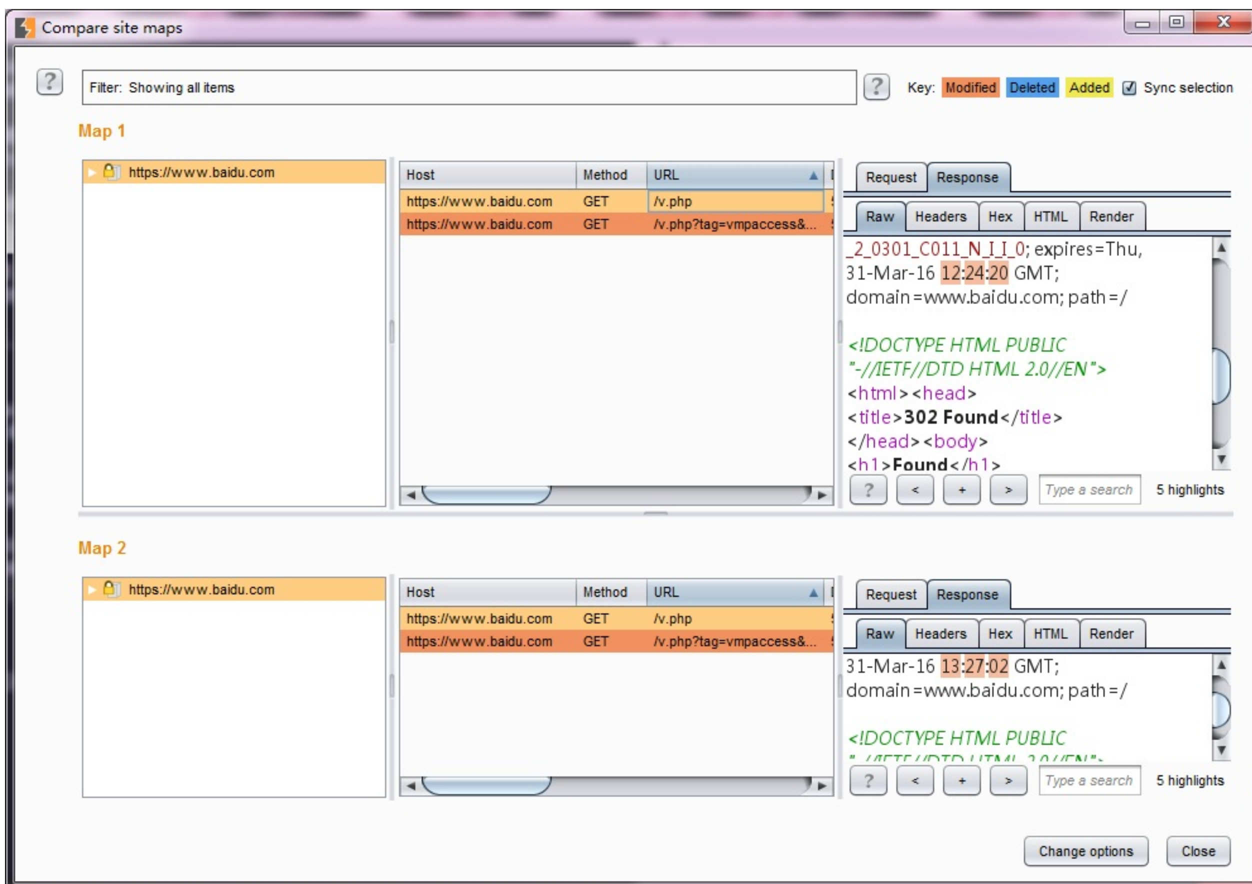
6.设置完Site Map 1 和Site Map 2之后，将进入请求消息匹配设置。在这个界面，我们可以通过URL文件路径、Http请求方式、请求参数、请求头、请求Body来对匹配条件进行过滤。



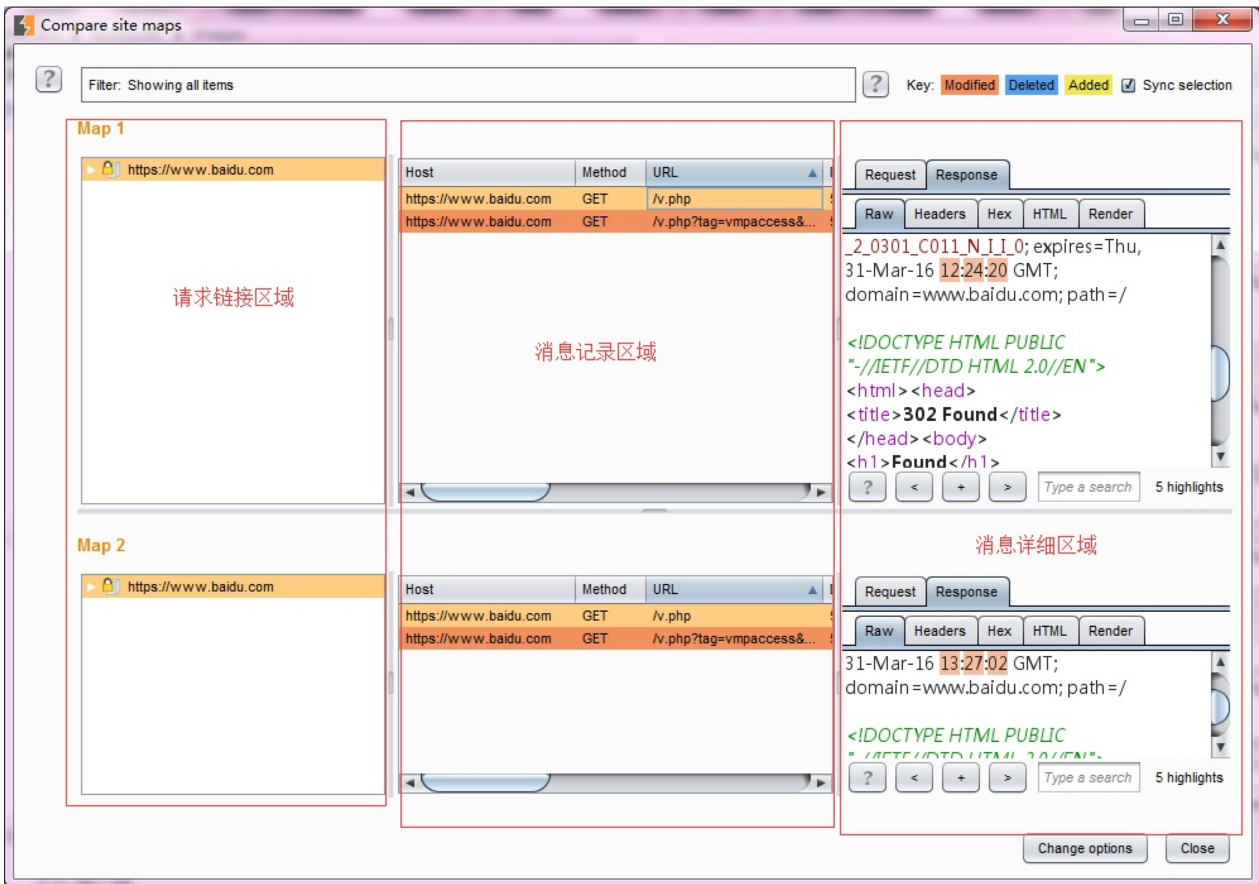
7..设置请求匹配条件，接着进入应答比较设置界面。在这个界面上，我们可以设置哪些内容我们指定需要进行比较的。从下图我们可以看出，主要有响应头、form表单域、空格、MIME类型。点击【Next】。



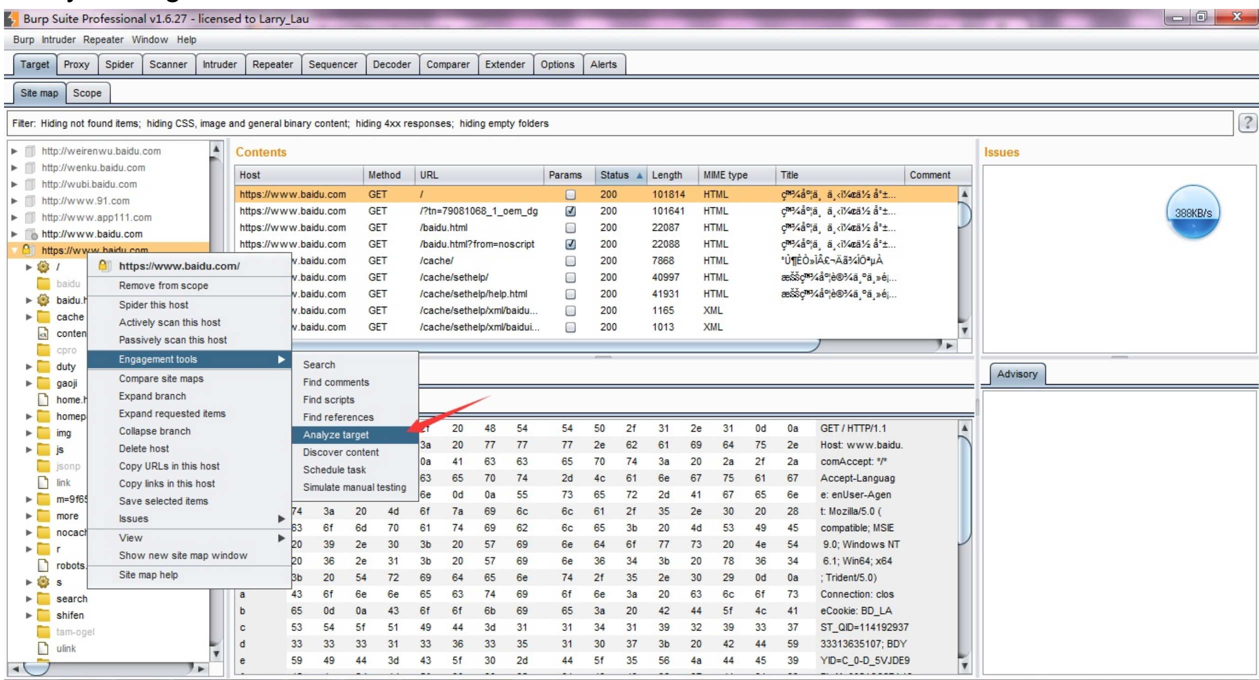
8.如果我们之前是针对全站进行比较，且是选择重新发生一次作为Site Map2的方式，则界面加载过程中会不停提示你数据加载的进度，如果涉及功能请求的链接较少，则很快进入比较界面。如下图。



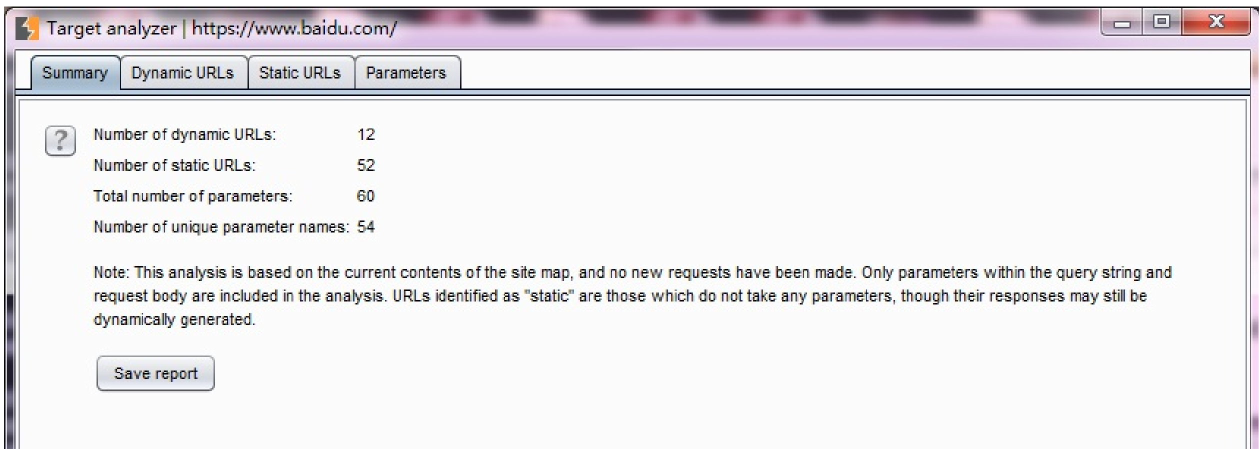
9.从上图我们可以看到，站点比较的界面上部为筛选过滤器（这个过滤器与其他过滤器使用雷同，此处不再赘述），下部由左、中、右三块构成。左边为请求的链接列表，中间为Site Map 1 和Site Map 2的消息记录，右边为消息详细信息。当我们选择Site Map 1某条消息记录时，默认会自动选择Site Map 2与之对应的记录，这是有右上角的【同步选择】勾选框控制的，同时，在右边的消息详细区域，会自动展示Site Map 1与Site Map 2通信消息的差异，包含请求消息和应答消息，存在差异的地方用底色标注出来。



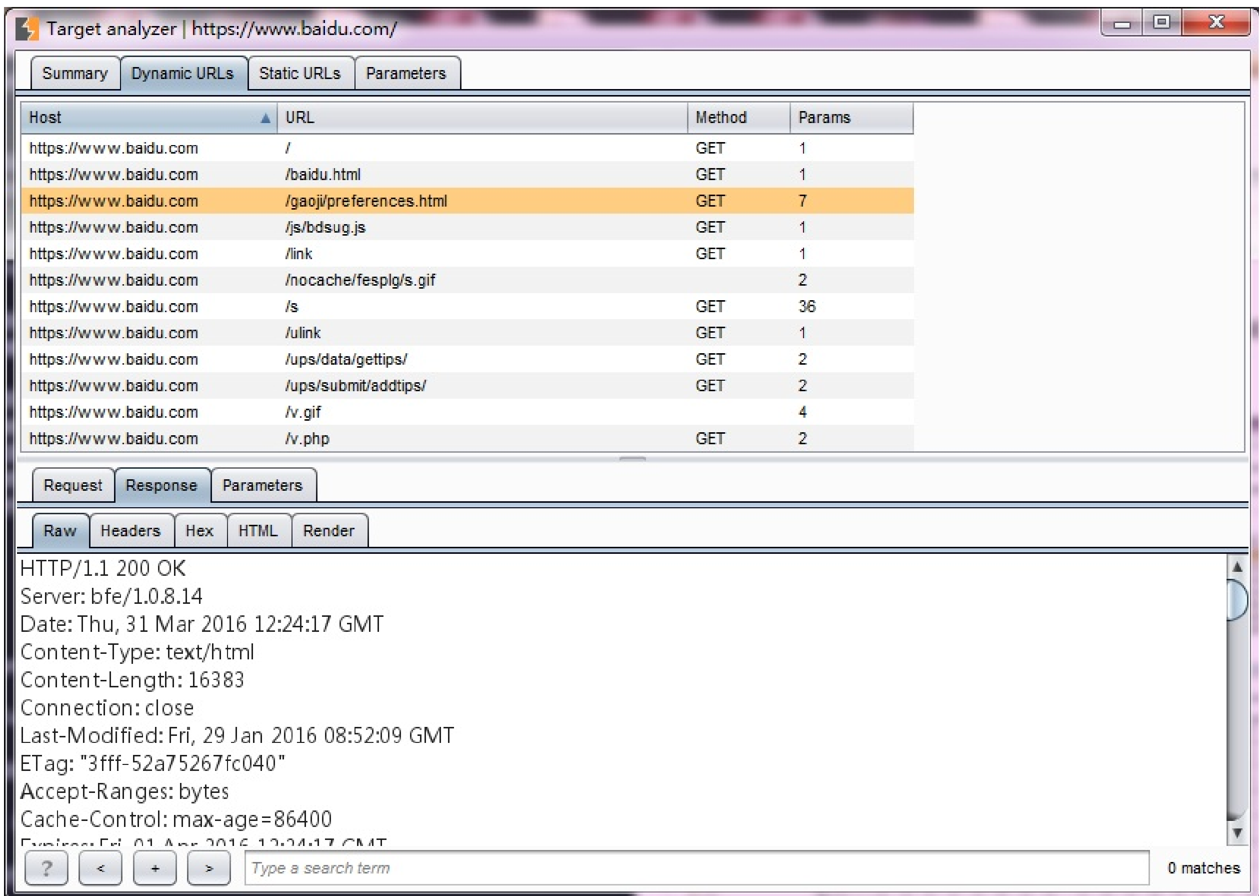
攻击面分析是Burp Suite 交互工具（Engagement tools）中的功能，这里我们先看看Analyze Target使用，其他的功能会在高级使用相关章节讲述。 1.首先，我们通过站点地图，打开Analyze Target，如图所示。



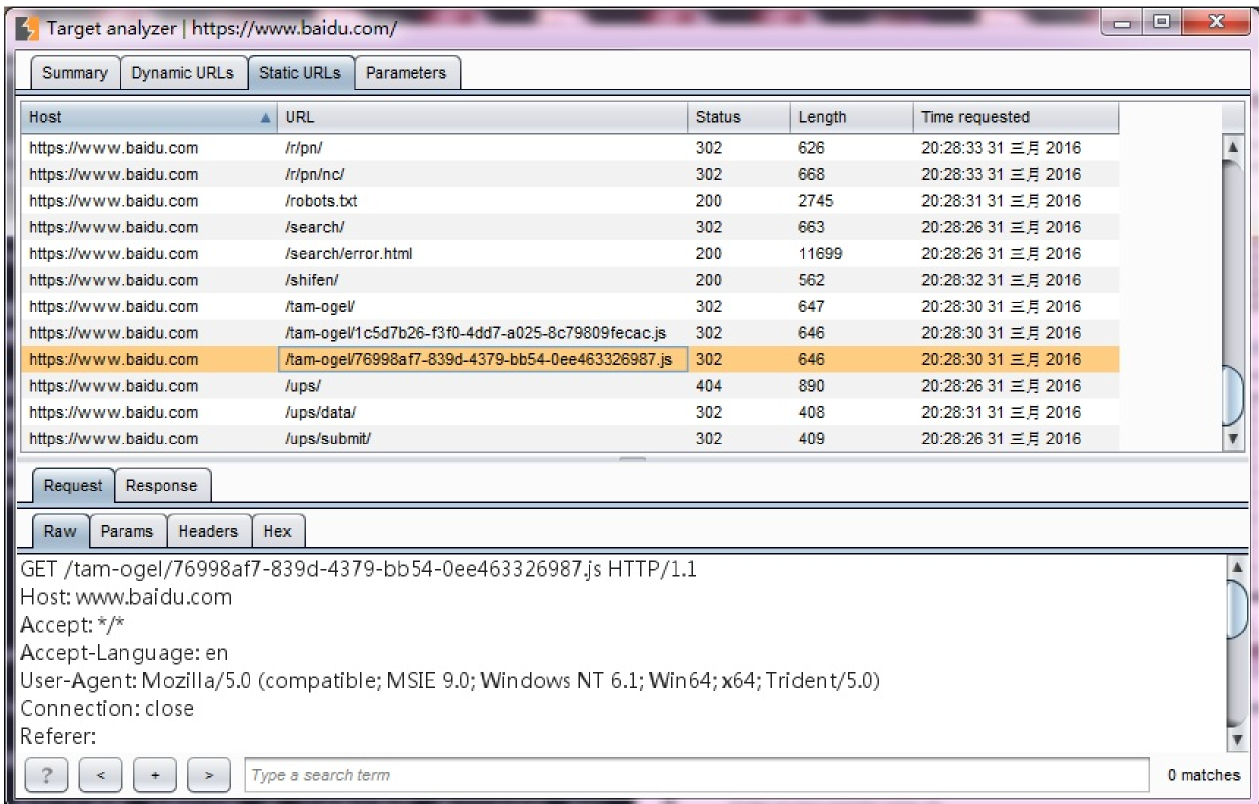
2.在弹出的分析界面中，我们能看到概况、动态URL、静态URL、参数4个视图。



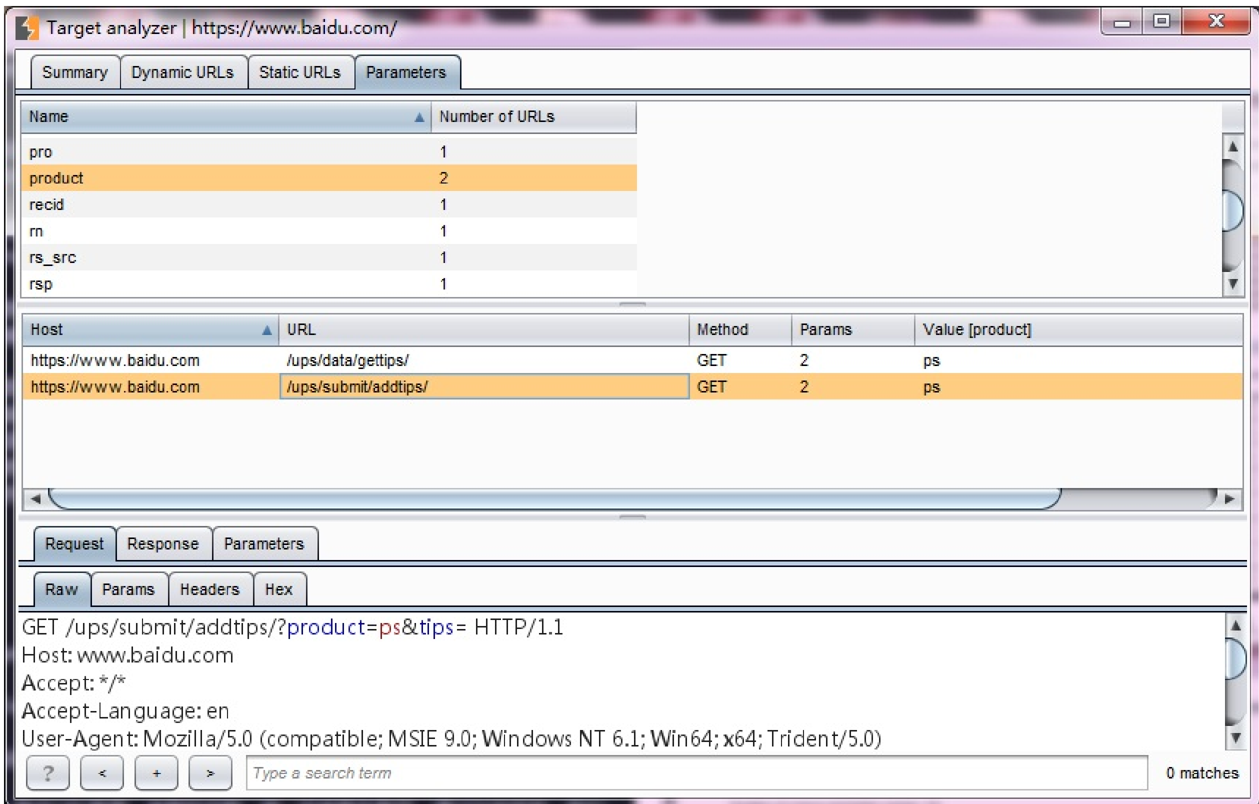
3.概况视图主要展示当前站点动态URL数量、静态URL数量、参数的总数、唯一的参数名数目，通过这些信息，我们对当前站点的总体状况有粗线条的了解。 4.动态URL视图展示所有动态的URL请求和应答消息，跟其他的工具类似，当你选中某一条消息时，下方会显示此消息的详细信息。



5.静态URL视图与动态URL视图类似，如图.



6.参数视图有上中下三部分组成，上部为参数和参数计数统计区，你可以通过参数使用的次数进行排序，对使用频繁的参数进行分析；中部为参数对于的使用情况列表，记录对于的参数每一次的使用记录；下部为某一次使用过程中，请求消息和应答消息的详细信息。



在使用攻击面分析功能时，需要注意，此功能主要是针对站点地图中的请求URL进行分析，如果某些URL没有记录，则不会被分析到。同时，在实际使用中，存在很点站点使用伪静态，如果请求的URL中不带有参数，则分析时无法区别，只能当做静态URL来分析。

第六章 如何使用Burp Spider

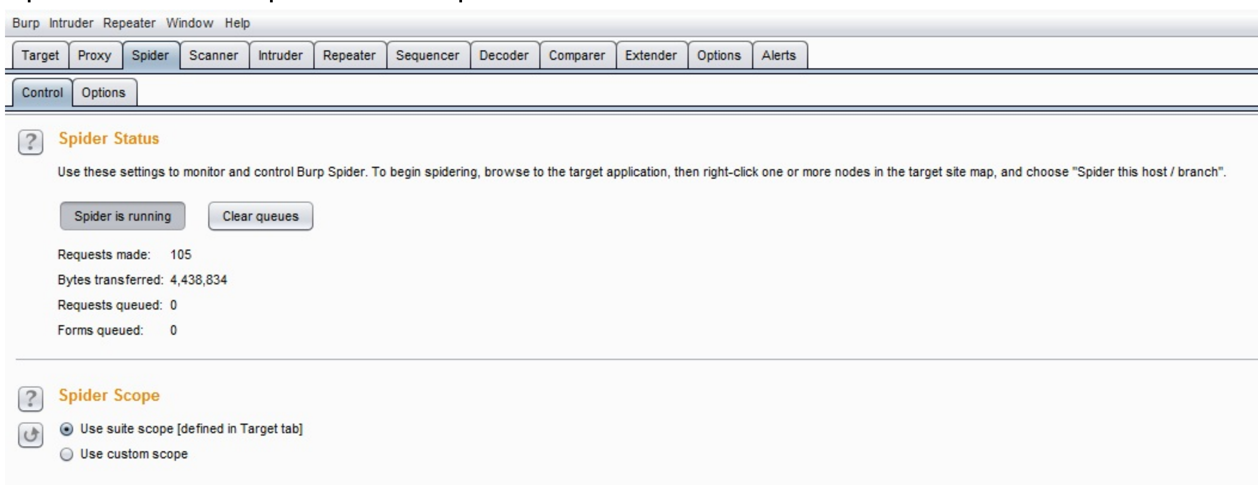
通过前一章的学习，我们了解到，存在于Burp Target中的站点信息，我们可以直接传送到Burp Spider中进行站点信息的爬取。这一章我们重点来学习Burp Spider的使用，主要包含两个方面：

- Spider控制（Control）
- Spider可选项设置（Options）

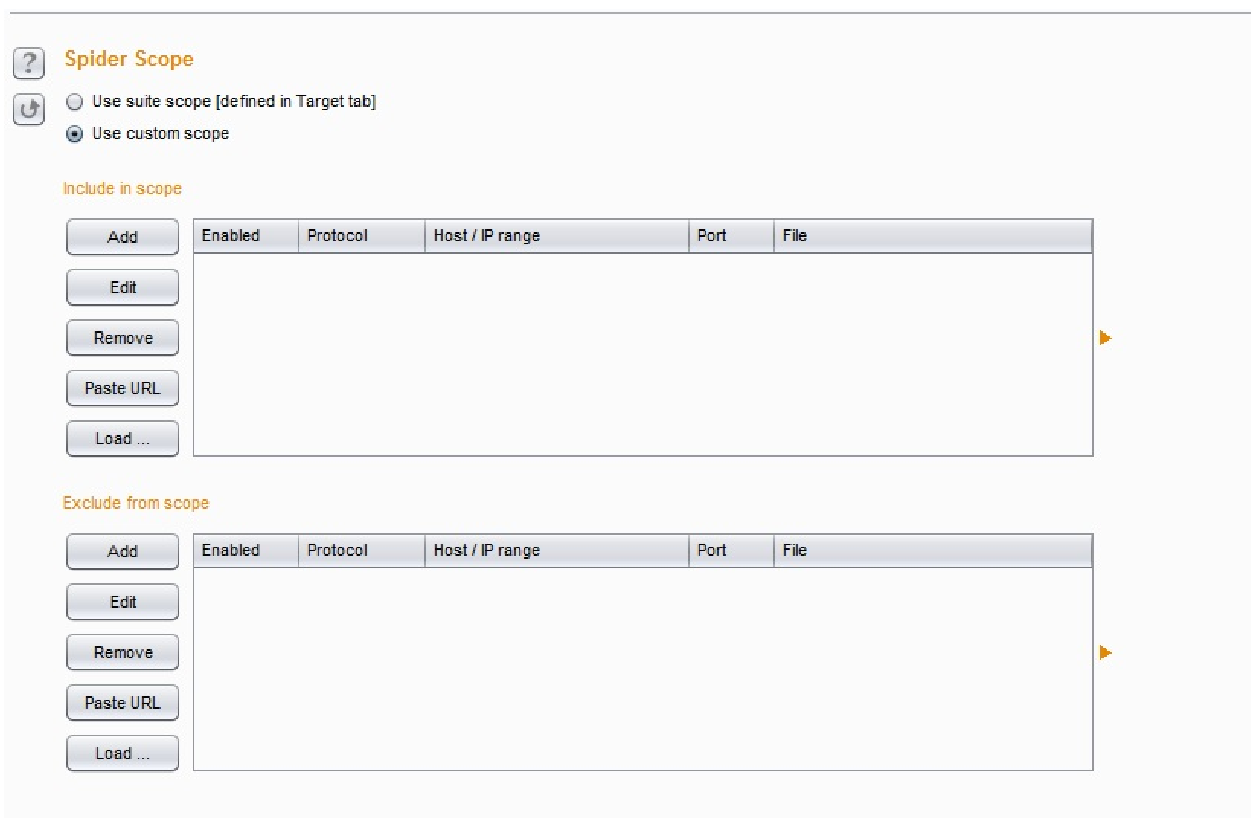
Burp Spider的功能主要使用于大型的应用系统测试，它能在很短的时间内帮助我们快速地了解系统的结构和分布情况，下面我们就先来看看Spider控制，

Spider控制

Spider控制界面由Spider 状态和Spider 作用域两个功能组成。



Spider 状态除了显示当前进度、传输情况、请求队列等统计信息外，还有Spider运行/暂停按钮与清空队列按钮，分别用来控制Spider是否运行和队列中的数据管理。而Spider 作用域是用来控制Spider的抓取范围，从图中我们可以看到有两种控制方式，一种是使用上一章讲的Target Scope，另一种是用户自定义。当我们选中用户自定义按钮，界面改变成下面的样子，如下图所示。

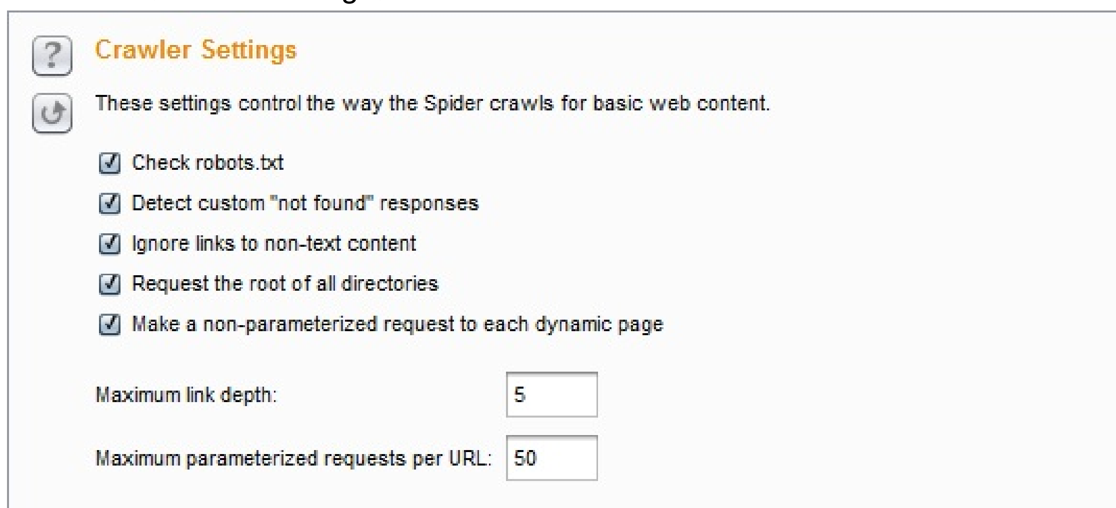


此处用户自定义作用域的配置与Target Scope 的配置完全一致，具体使用方法请参数上一章Target Scope 的配置。

Spider可选项设置

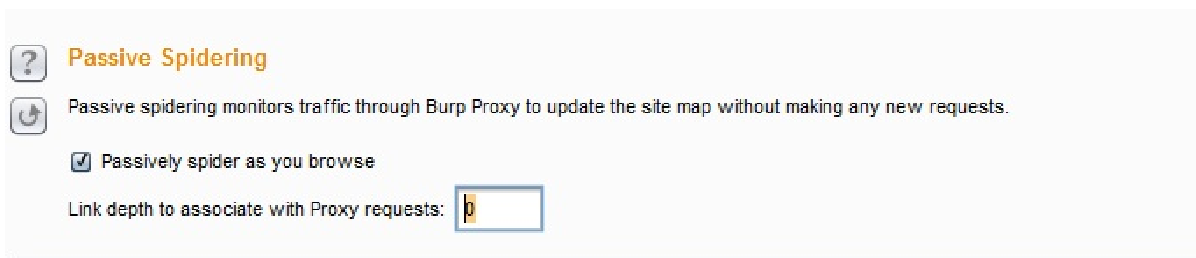
Spider可选项设置由抓取设置、抓取代理设置、表单提交设置、应用登陆设置、蜘蛛引擎设置、请求消息头设置六个部分组成。

- 抓取设置（Crawls Settings） -此项是用来控制蜘蛛抓取网页内容的方式



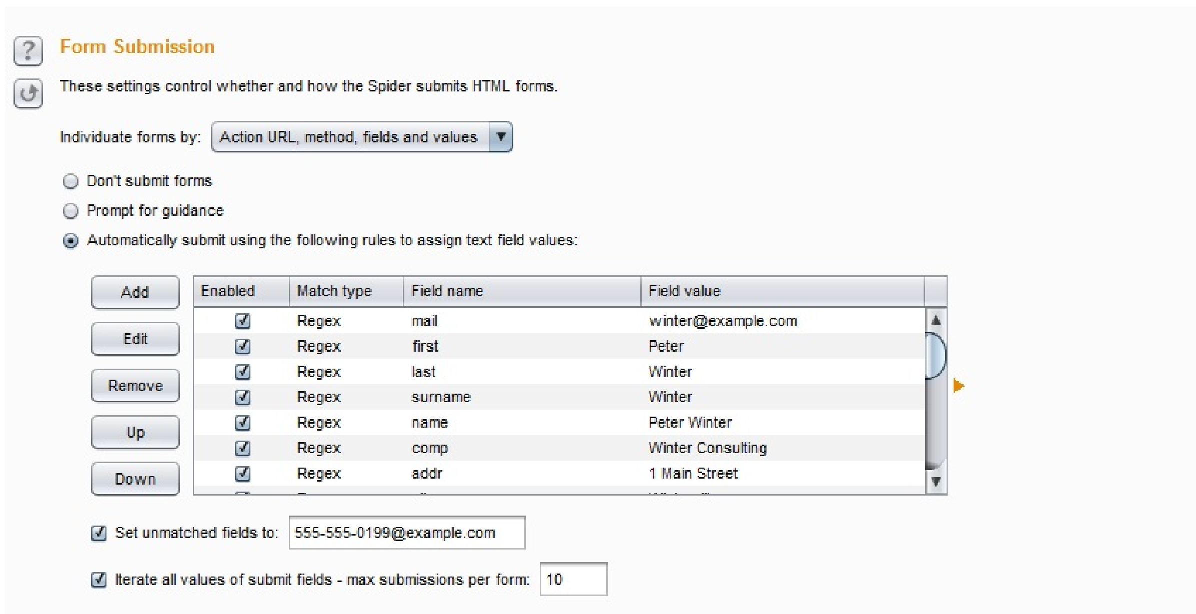
自上而下依次是：检查robots.txt 文件、检测404应答、忽略内容为空的链接、爬取根目录下所有文件和目录、对每一个动态页面发送无参数请求、最大链接深度、最大请求URL参数数目

- 抓取代理设置（Passive Spidering ）

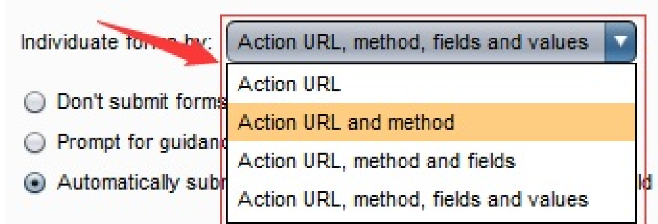


这个设置比较简单，第一个如果勾选，则爬取时通过Burp Proxy，反之则不通过。第二个设置是控制代理的链接深度。默认为0，表示无限深度，即无论有多少层级的URL均需要爬取。

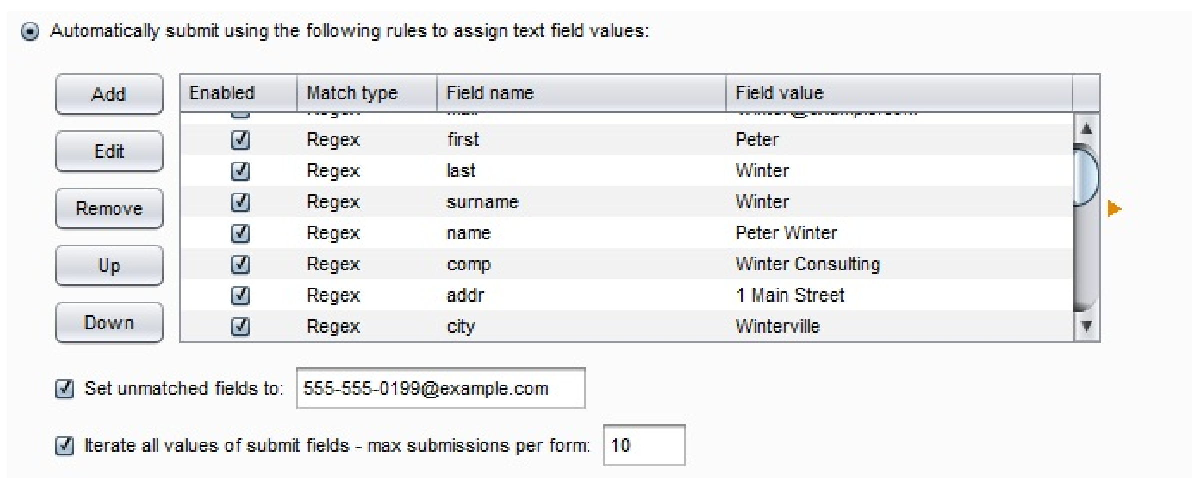
- 表单提交设置（Form Submission） 表单提交设置主要是用来控制在蜘蛛抓取过程中，对于form表单的处理方式，其界面如下图：



第一个下拉选项中，是对form表单域的处理内容做控制，默认选择Action URL、method、fields、values，即同时处理请求的url、请求方式GET或者POST、包含哪些属性名以及属性值。点击下拉选项，可以选择其中一个或者几个。如下图：

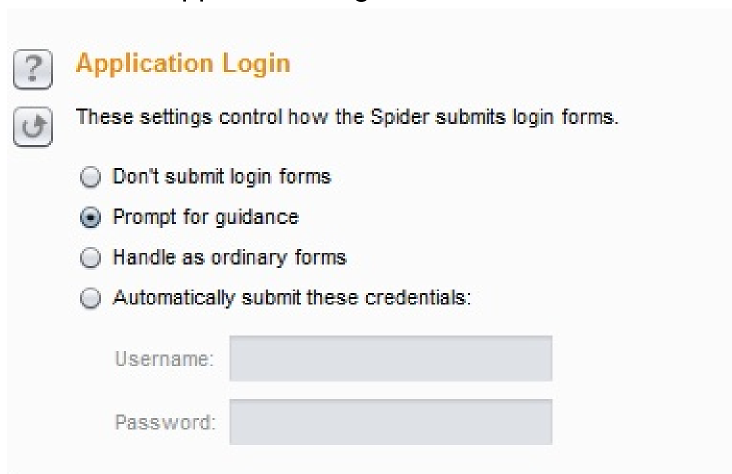


接下来的设置的控制form表单的处理方式：不提交表单、需要手工确认、使用默认值自动填写三种方式。 不提交表单的含义是抓取时候不提交表单数据，这个非常好理解；需要手工确认是指当抓取表单时，弹出界面，让渗透测试人员自己手工确认表单数据；使用默认值自动填写是对表单的内容，使用下方的各个配置项进行匹配（匹配时可以使用完全匹配和正则表达式匹配两种方式其一），默认填写这些值，然后自动进行提交。其界面如下图所示：



从上图我们可以看出，对于表单的输入域我们可以添加和修改以满足实际情况的需要，如果还有其他的属性输入域我们不想每一个都录入，可以勾选“设置不匹配的属性值”，统一指定输入的值。如图中的555-555-0199@example.com

- 应用登陆（Application Login） 此选择项主要用来控制抓取时，登陆页面的处理方式。



选择项依次是：不提交登陆信息、手工确认登陆信息、作为普通表单处理（如果选择此项，则把登陆表单的form当作其他表单一样处理，对于登陆表单将使用"表单提交设置" 中的具体配置）、自动提交登陆（选择此项，需要在下方的输入框中指定用户名和密码）

- 蜘蛛引擎设置（Spider Engine)和HTTP 消息头设置（Requests Header）

Spider Engine

These settings control the engine used for making HTTP requests when spidering.

Number of threads: 10

Number of retries on network failure: 3

Pause before retry (milliseconds): 2000

☐ Throttle between requests (milliseconds): 0

☐ Add random variations to throttle

Request Headers

These settings control the request headers used in HTTP requests made by the Spider.

Add | Edit | Remove | Up | Down

Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64; Trident/5.0)
Connection: close

☑ Use HTTP version 1.1

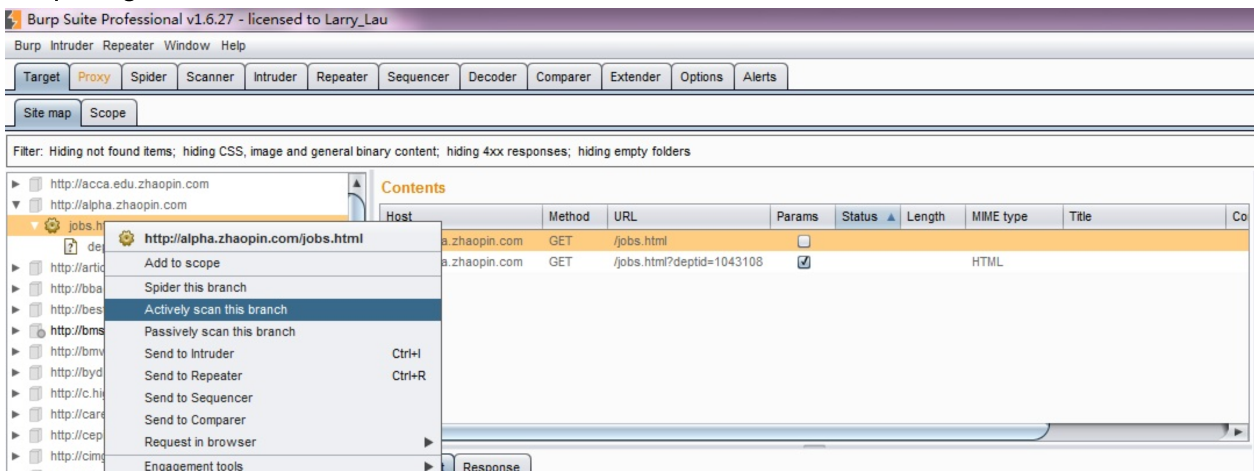
☑ Use Referer header

其中蜘蛛引擎设置主要是用来控制蜘蛛抓取的线程数、网络失败时重试的次数、重试暂停间隙等，而HTTP 消息头设置是用来设置Http请求的消息头自定义，比如说，我们可以编辑消息头信息，可以指定请求为移动设备，或者不同的手机型号，或者指定为Safari浏览器，指定HTTP协议版本为1.1、使用referer等。

第七章 如何使用Burp Scanner

Burp Scanner的功能主要是用来自动检测web系统的各种漏洞，我们可以使用Burp Scanner代替我们手工去对系统进行普通漏洞类型的渗透测试，从而能使得我们把更多的精力放在那些必须要人工去验证的漏洞上。

在使用Burp Scanner之前，我们除了要正确配置Burp Proxy并设置浏览器代理外，还需要在Burp Target的站点地图中存在需要扫描的域和URL模块路径。如下图所示：



当Burp Target的站点地图中存在这些域或URL路径时，我们才能对指定的域或者URL进行全扫描或者分支扫描。下面我们就来整体的学习一下，一次完整的Burp Scanner使用大概需要哪些步骤。

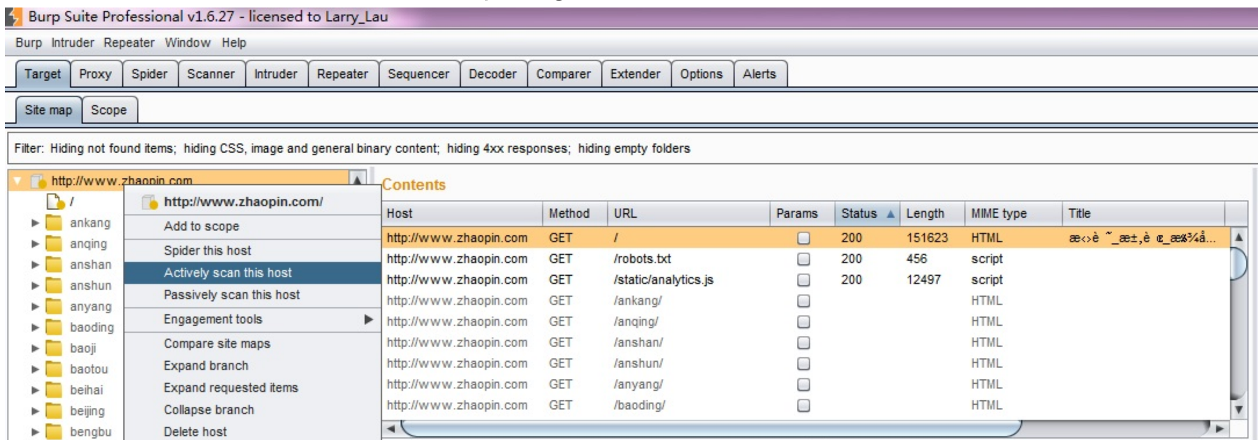
本章的主要内容有：

- Burp Scanner基本使用步骤
- Burp Scanner扫描方式
- Burp Scanner扫描报告
- Burp Scanner扫描控制
- Burp Scanner可选项设置

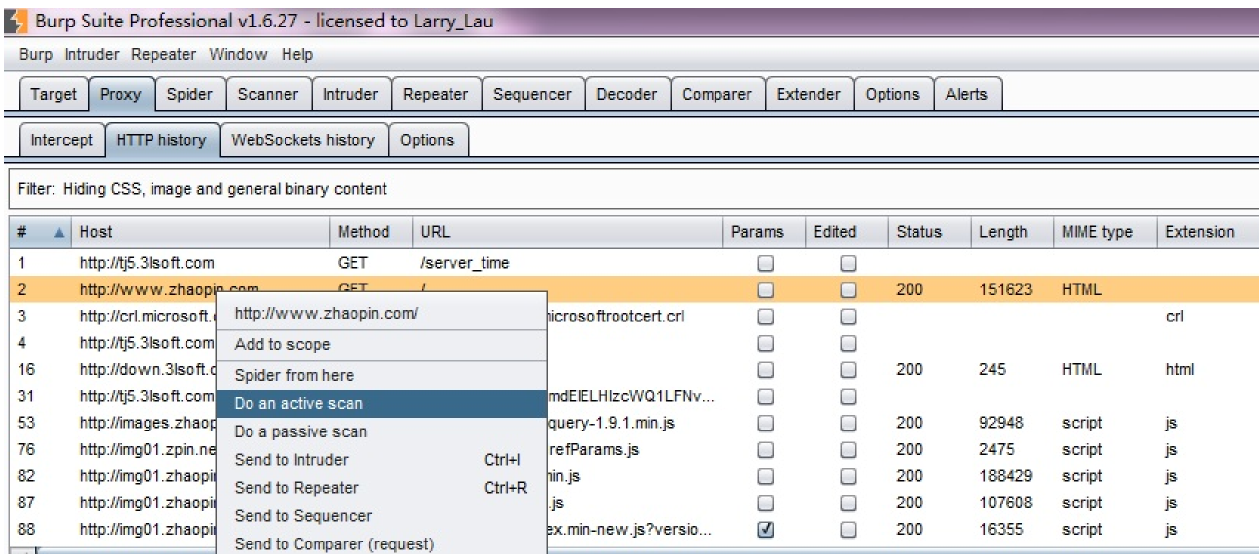
Burp Scanner基本使用步骤

Burp Scanner基本使用主要分为以下15个步骤，在实际使用中可能会有所改变，但大体的环节主要就是下面的这些。 1.确认Burp Suite正常启动并完成浏览器代理的配置。 2.进入Burp Proxy，关闭代理拦截功能，快速的浏览需要扫描的域或者URL模块。 3.当我们浏览时，默认情况下，Burp Scanner会扫描通过代理服务的请求，并对请求的消息进行分析来辨别是非存在系统漏洞。同时，当我们打开Burp Target时，也会在站点地图中显示请求的URL树。

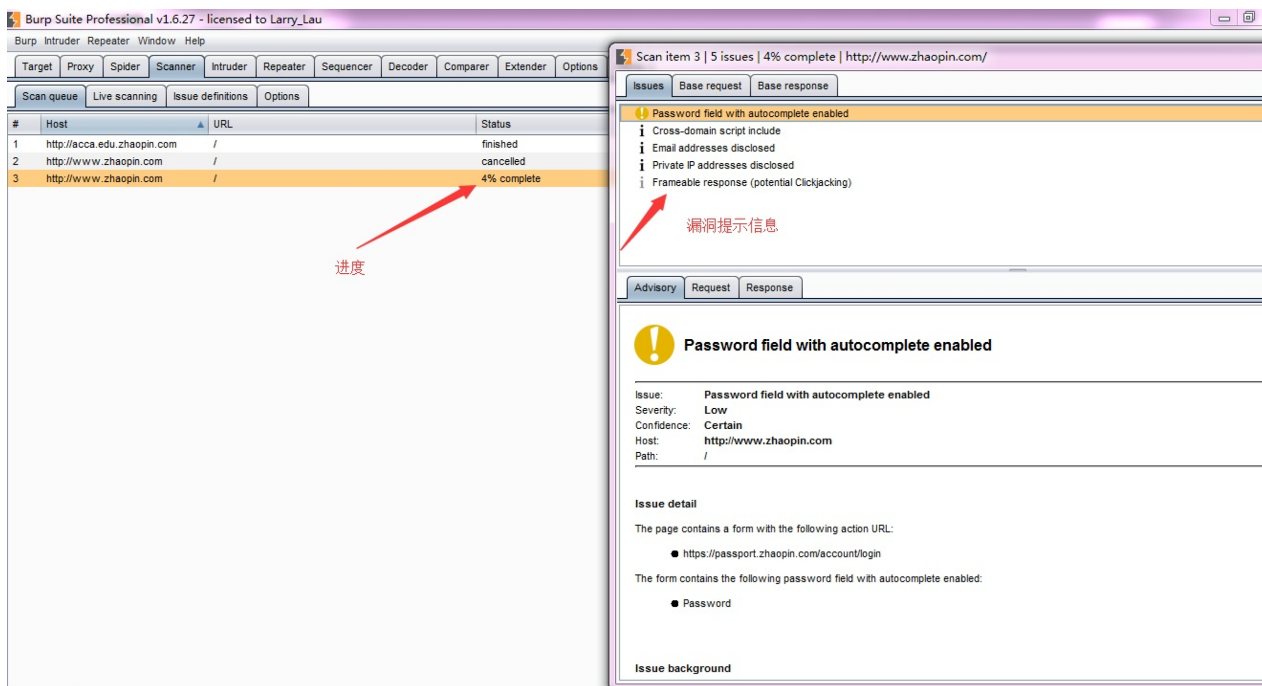
Burp Scanner基本使用主要分为以下15个步骤，在实际使用中可能会有所改变，但大体的环节主要就是下面的这些。 1.确认Burp Suite正常启动并完成浏览器代理的配置。 2.进入Burp Proxy，关闭代理拦截功能，快速的浏览需要扫描的域或者URL模块。 3.当我们浏览时，默认情况下，Burp Scanner会扫描通过代理服务的请求，并对请求的消息进行分析来辨别是非存在系统漏洞。同时，当我们打开Burp Target时，也会在站点地图中显示请求的URL树。



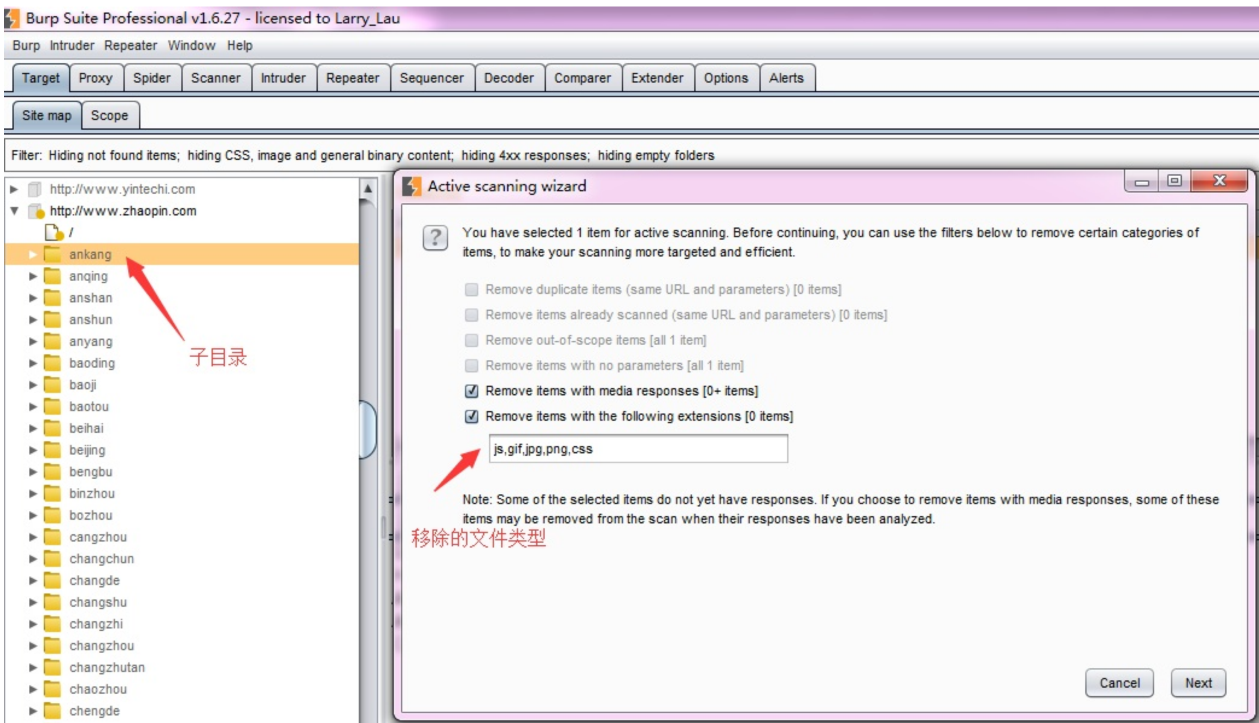
4.我们可以有针对性的选择Burp Target站点地图下的某个节点上链接URL上，弹出右击菜单，进行Active Scan。然后在弹出的确认框中，点击【YES】即进行扫描整个域。



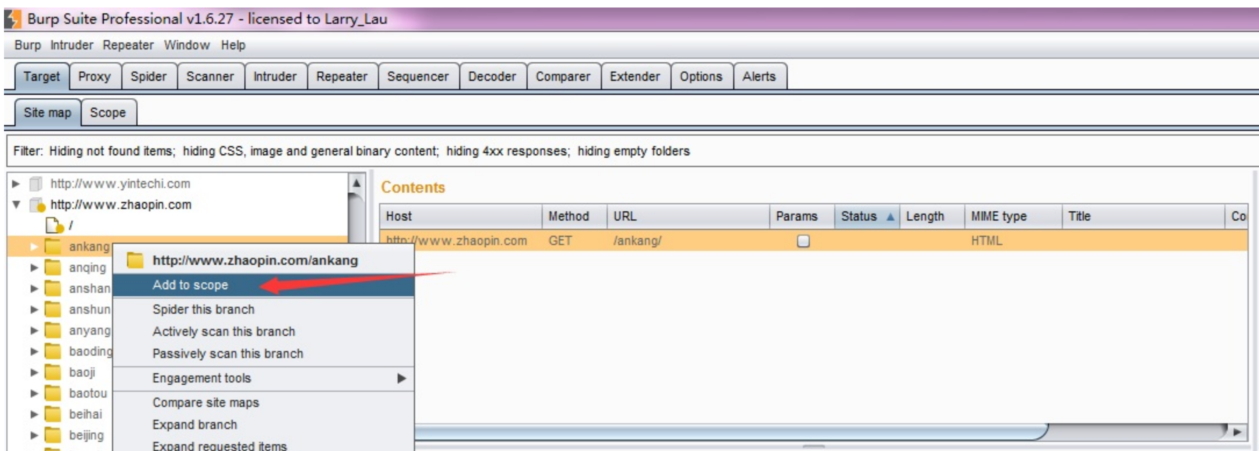
6.这时，我们打开Burp Scanner 选项卡，在队列子选项卡中，会看到当前扫描的进度。如果我们双击URL，则弹出扫描结果的提示信息。



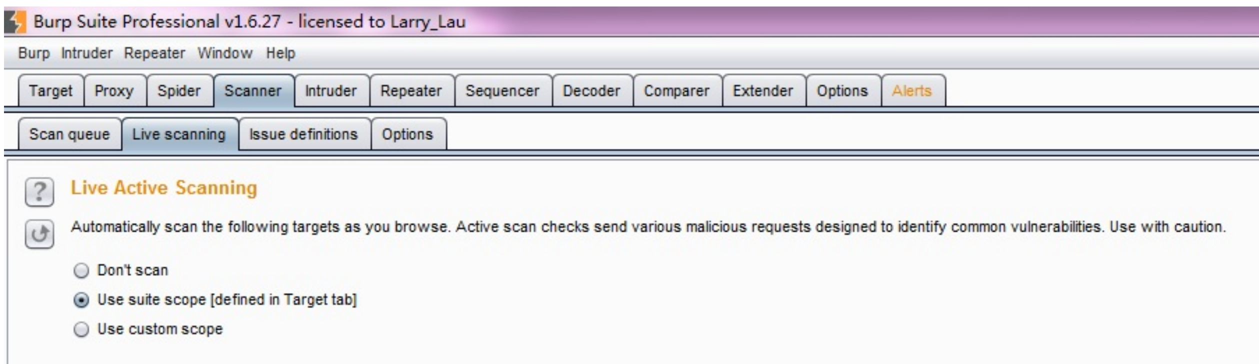
7.如果我们在Burp Target站点地图下选择某个子目录进行扫描，则会弹出更优化的扫描选项，我们可以对选项进行设置，指定哪些类型的文件不再扫描范围之内。



8.当我们再次返回到Burp Scanner 选项卡界面时，选择的子目录已经开始在扫描中，其扫描的进度依赖于需要扫描内容的多少。 9.如果我们没有定义了目标作用域（Target Scope），最简单的方式就是在Burp Target站点地图上右击弹出菜单中添加到作用域，然后自动进行扫描。

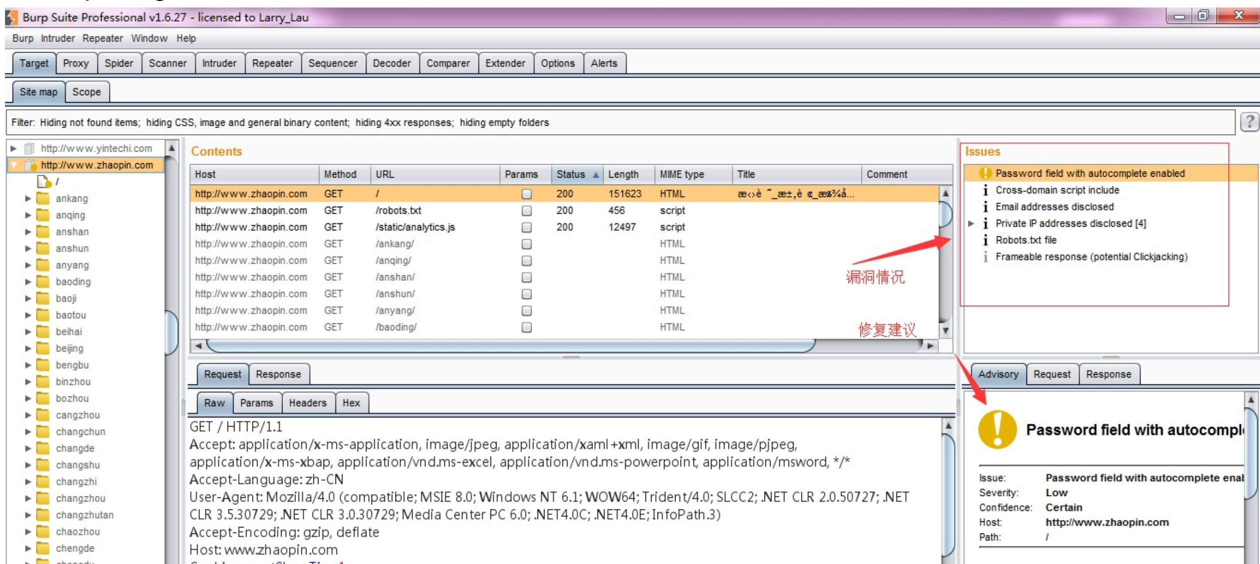


10.然后进入Burp Scanner的Live scanning子选项卡，在Live Active Scanning控制块中，选择Use suite scope，这样，Burp Scanner将自动扫描经过Burp Proxy的交互信息。

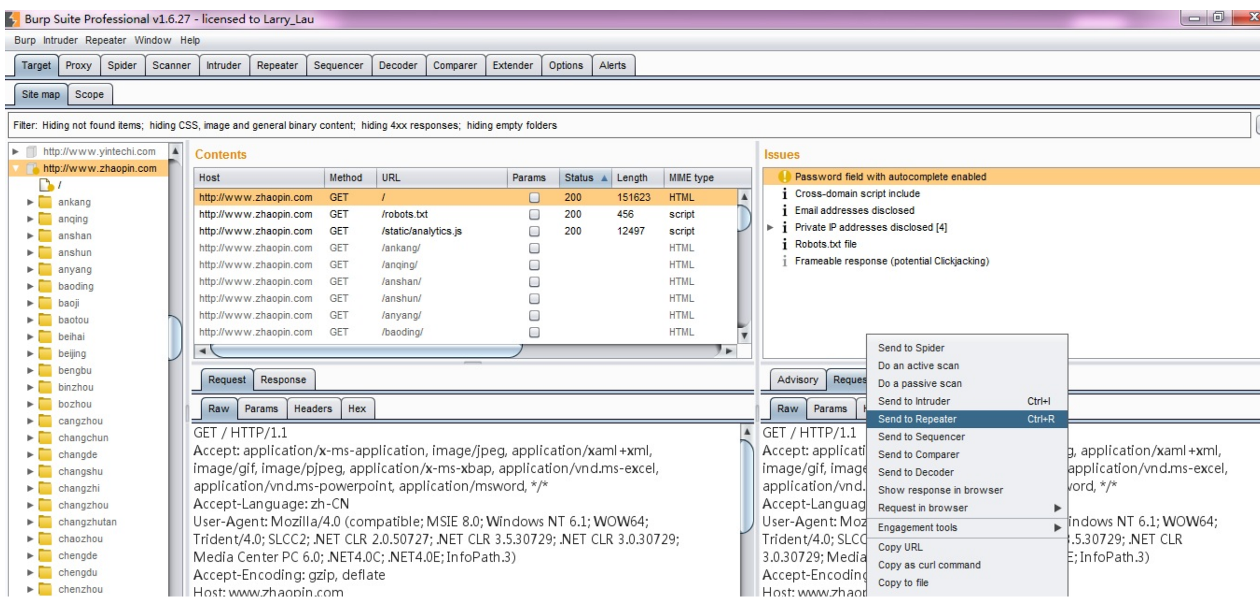


11.当我们再次使用浏览器对需要测试的系统进行浏览时，Burp Scanner不会发送额外的请求信息，自动在浏览的交互信息的基础上，完成对请求消息的漏洞分析。 12.此时，当我再返回

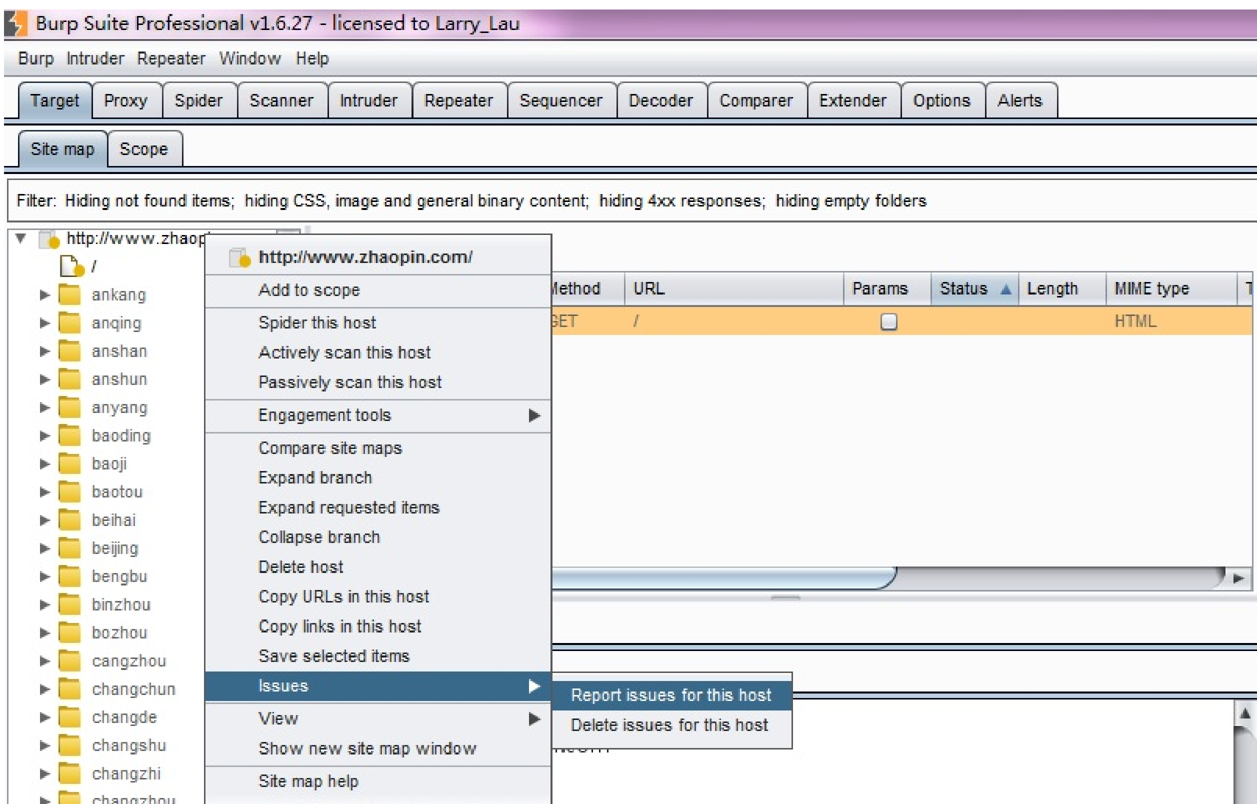
到Burp Target站点地图界面，将提示系统可能存在的漏洞情况，以及处理这些漏洞的建议。



13.同时，我们也可以在漏洞提示的请求信息上，将消息发送到Burp Repeater模块，对漏洞进行分析和验证。



14.随着Burp Scanner扫描的进度，在Burp Target站点地图界面上的issues模块中的漏洞信息也会不断的更新。 15.当Burp Scanner扫描完成之后，我们在Burp Target站点地图的选择链接右击，依次选择issues-->report issues for this host 即可导出漏洞报告。



Burp Scanner扫描方式

通过以上的操作步骤我们可以学习到，Burp Scanner扫描方式主要有两种：主动扫描和被动扫描

- 主动扫描（Active Scanning）

当使用主动扫描模式时，Burp 会向应用发送新的请求并通过payload验证漏洞。这种模式下的操作，会产生大量的请求和应答数据，直接影响系统的性能，通常使用在非生产环境。它对下列的两类漏洞有很好的扫描效果：

1. 客户端的漏洞，像XSS、Http头注入、操作重定向；
2. 服务端的漏洞，像SQL注入、命令行注入、文件遍历。

对于第一类漏洞，Burp在检测时，会提交一下input域，然后根据应答的数据进行解析。在检测过程中，Burp会对基础的请求信息进行修改，即根据漏洞的特征对参数进行修改，模拟人的行为，以达到检测漏洞的目的。 对于第二类漏洞，一般来说检测比较困难，因为是发生在服务器侧。比如说SQL注入，有可能是返回数据库错误提示信息，也有可能是什么也不反馈。Burp在检测过程中，采用各个技术来验证漏洞是否存在，比如诱导时间延迟、强制修改Boolean值，与模糊测试的结果进行比较，已达到高准确性的漏洞扫描报告。

- 被动扫描（Passive Scanning）

当使用被动扫描模式时，Burp不会重新发送新的请求，它只是对已经存在的请求和应答进行分析，这对系统的检测比较安全，尤其在你授权访问的许可下进行的，通常适用于生成环境的检测。一般来说，下列这些漏洞在被动模式中容易被检测出来：

1. 提交的密码为未加密的明文。
2. 不安全的Cookie的属性，比如缺少的HttpOnly和安全标志。
3. cookie的范围缺失。
4. 跨域脚本包含和站点引用泄漏。
5. 表单值自动填充，尤其是密码。
6. SSL保护的内容缓存。
7. 目录列表。
8. 提交密码后应答延迟。
9. session令牌的不安全传输。
10. 敏感信息泄露，像内部IP地址，电子邮件地址，堆栈跟踪等信息泄漏。
11. 不安全的ViewState的配置。
12. 错误或者不规范的Content-type指令。

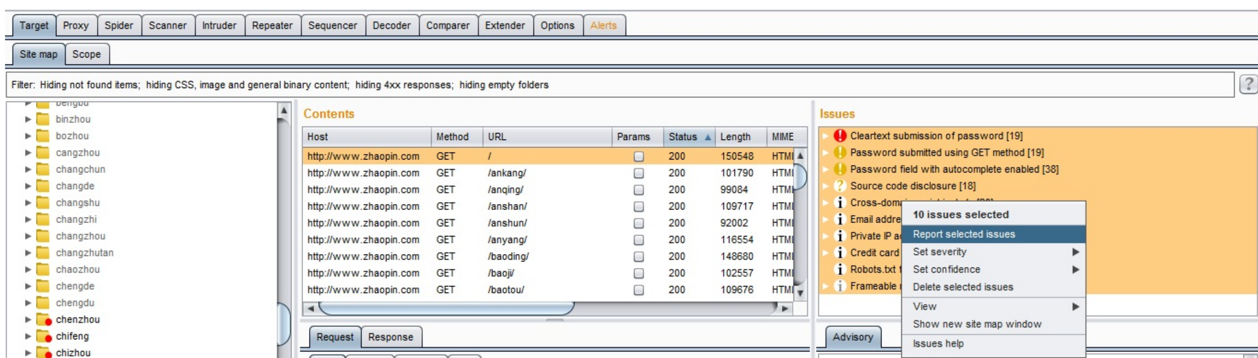
虽然被动扫描模式相比于主动模式有很多的不足，但同时也具有主动模式不具备的优点，除了前文说的对系统的检测在我们授权的范围内比较安全外，当某种业务场景的测试，每测试一次都会导致业务的某方面问题时，我们也可以使用被动扫描模式，去验证问题是否存在，减少测试的风险。

Burp Scanner扫描报告

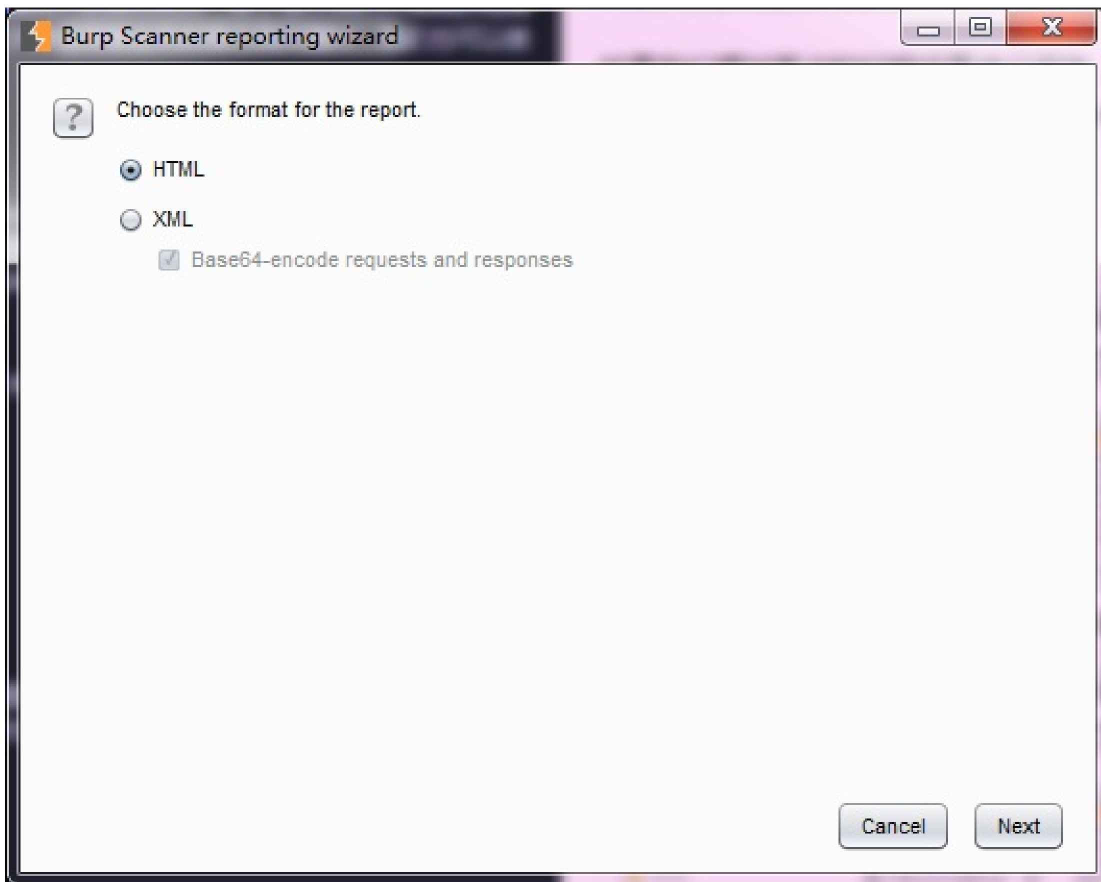
当我们对一个系统进行扫描完毕后，通常需要生成扫描报告，Burp Scanner支持的报告类型有HTML和XML两种格式。无法何种格式的扫描报告，其内容基本一致，主要由以下部分组成。报告样例可以点击Burp Scanner report查看.

除了头部的综述和目录外，每一个漏洞的章节通常包含：1.序号 表示漏洞的序号，如果有多个同样的漏洞，报告中只会有一个序号。2.漏洞的类型，可以近似地理解与OWASP的类型相对应。3.漏洞名称，具体可参考 Issue Definitions子选项卡。4.漏洞路径，漏洞对应的多个URL链接。5.漏洞的发生点，通常为参数名。6.问题的描述（Issue background） 描述漏洞发生的成因 7.解决建议（Remediation background）提供解决的思路和建议 8.请求消息和应答消息的详细信息。

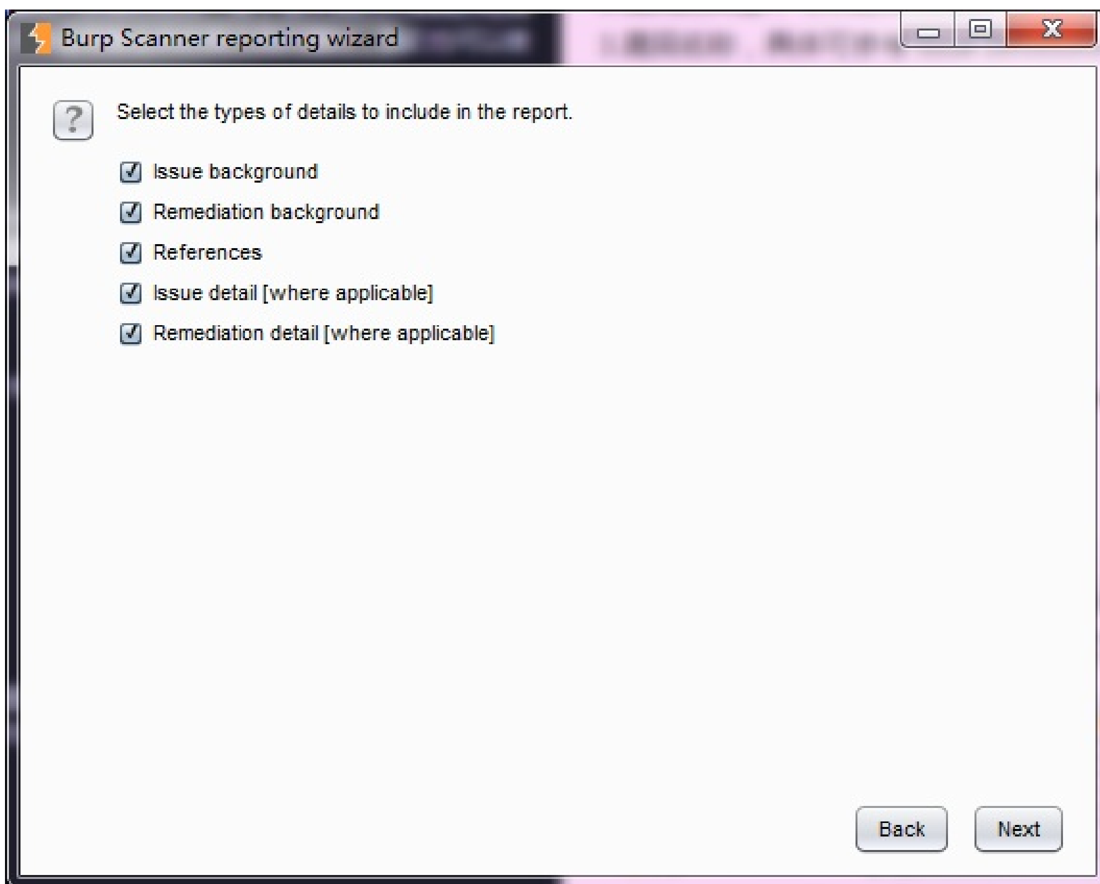
如果我们想对某次的扫描结果进行保存，需要Burp Target 的站点地图子选项卡的问题面板（Issue）上右击，在弹出的菜单中选择report Issues进行设置并保存即可。（注意，如果想导出所有的漏洞，需要选中所有的问题列表） 具体导出漏洞报告的步骤如下：1.选中需要保存的漏洞，右击弹出菜单，如下图：



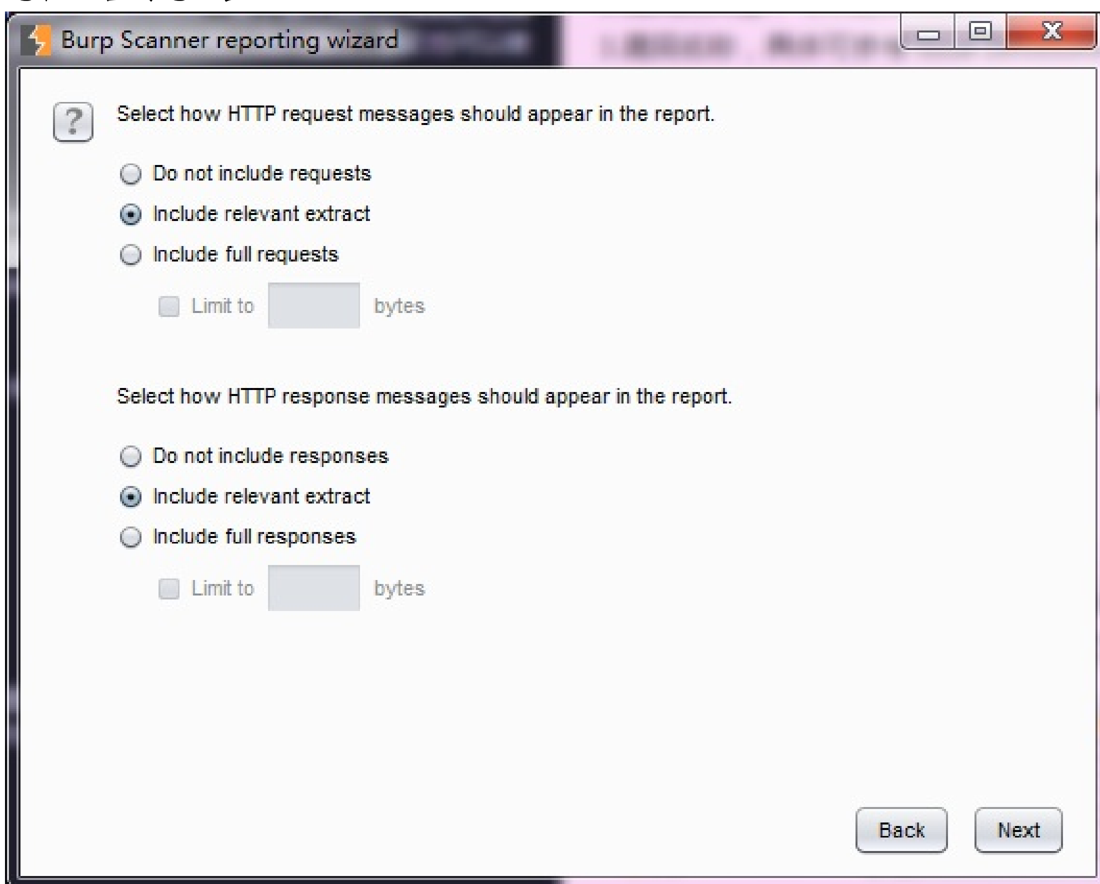
2.在弹出的对话框中选择需要保存的漏洞报告格式。



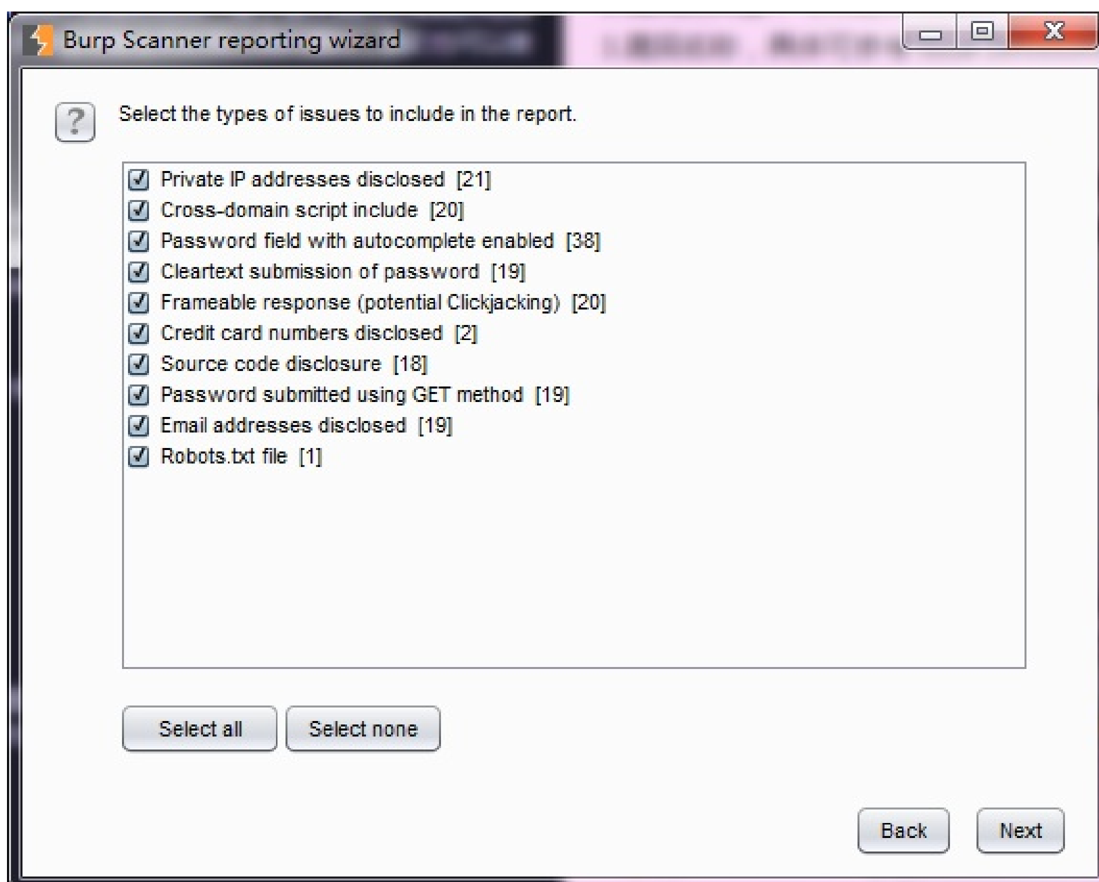
3.选择漏洞明细包含内容。



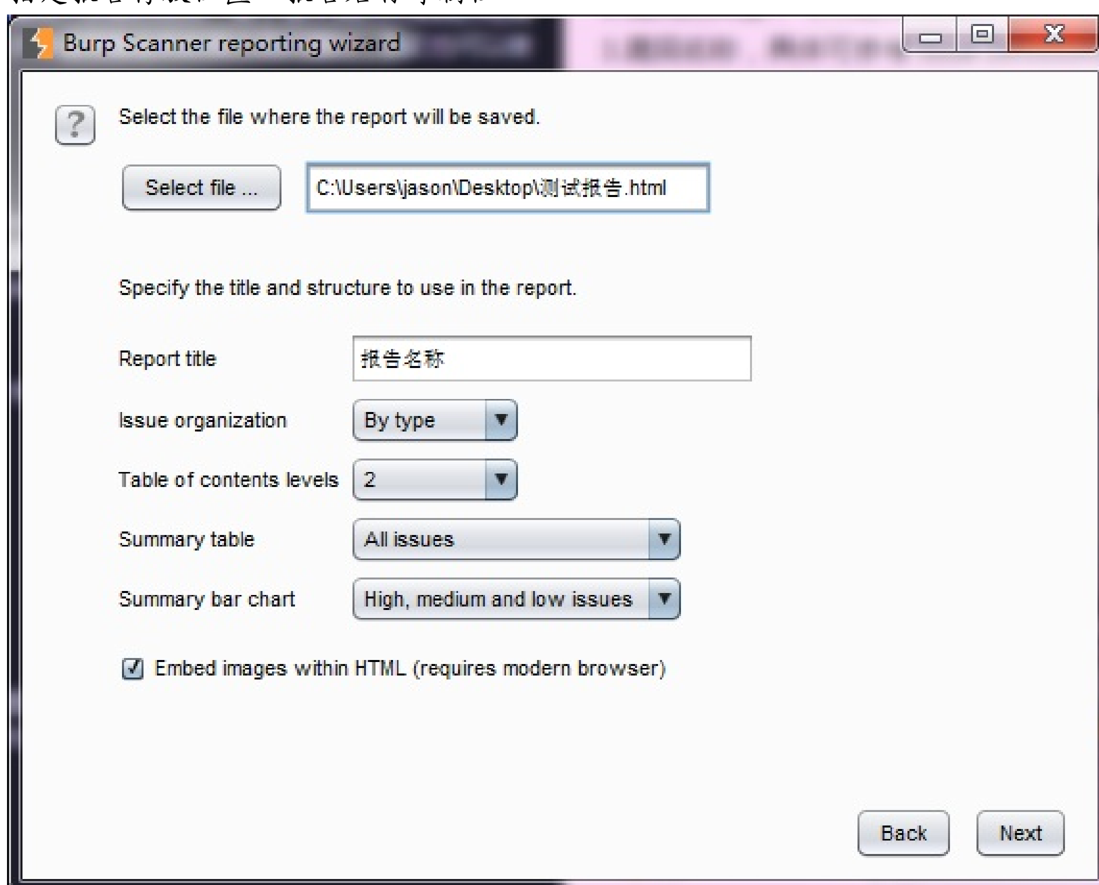
4.请求消息和应答消息设置。



5.选择报告包含的哪些漏洞。

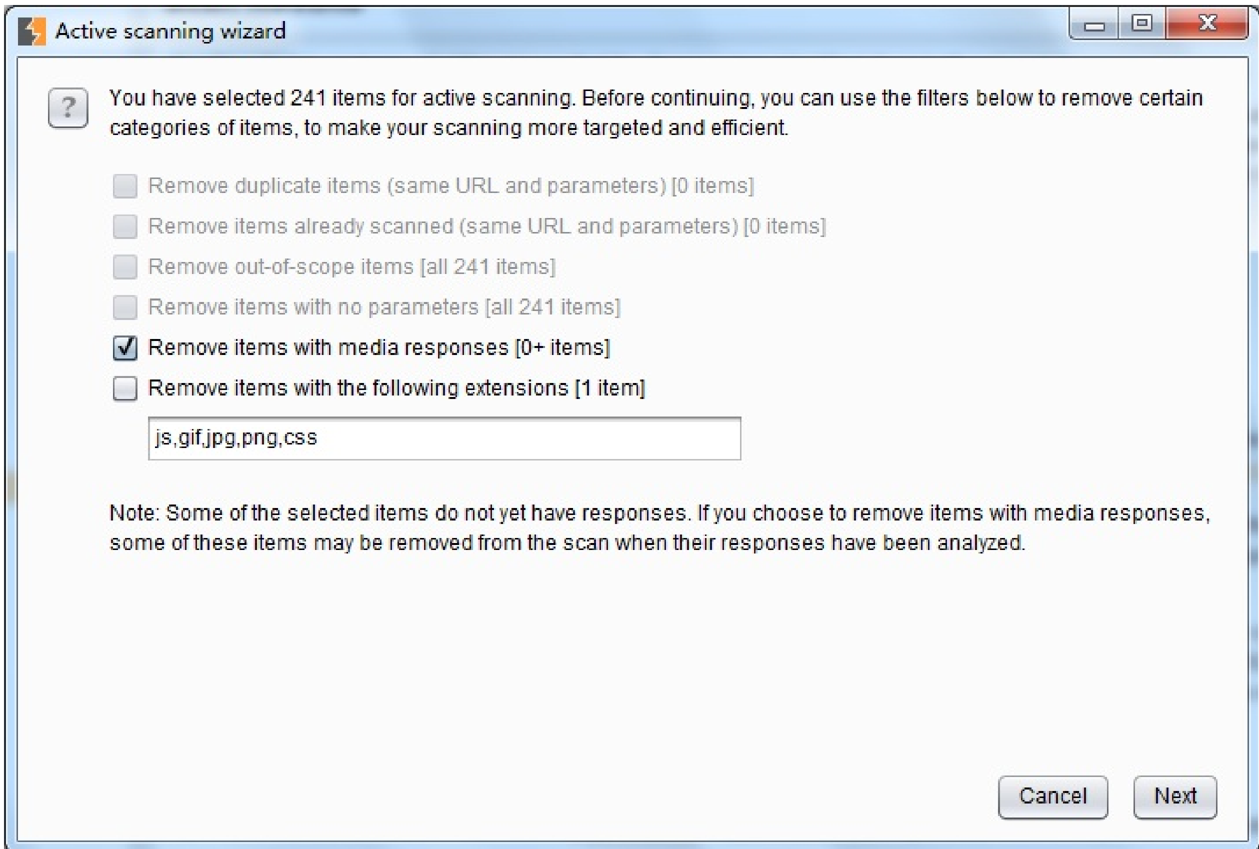


6.最后，指定报告存放位置、报告名称等属性。

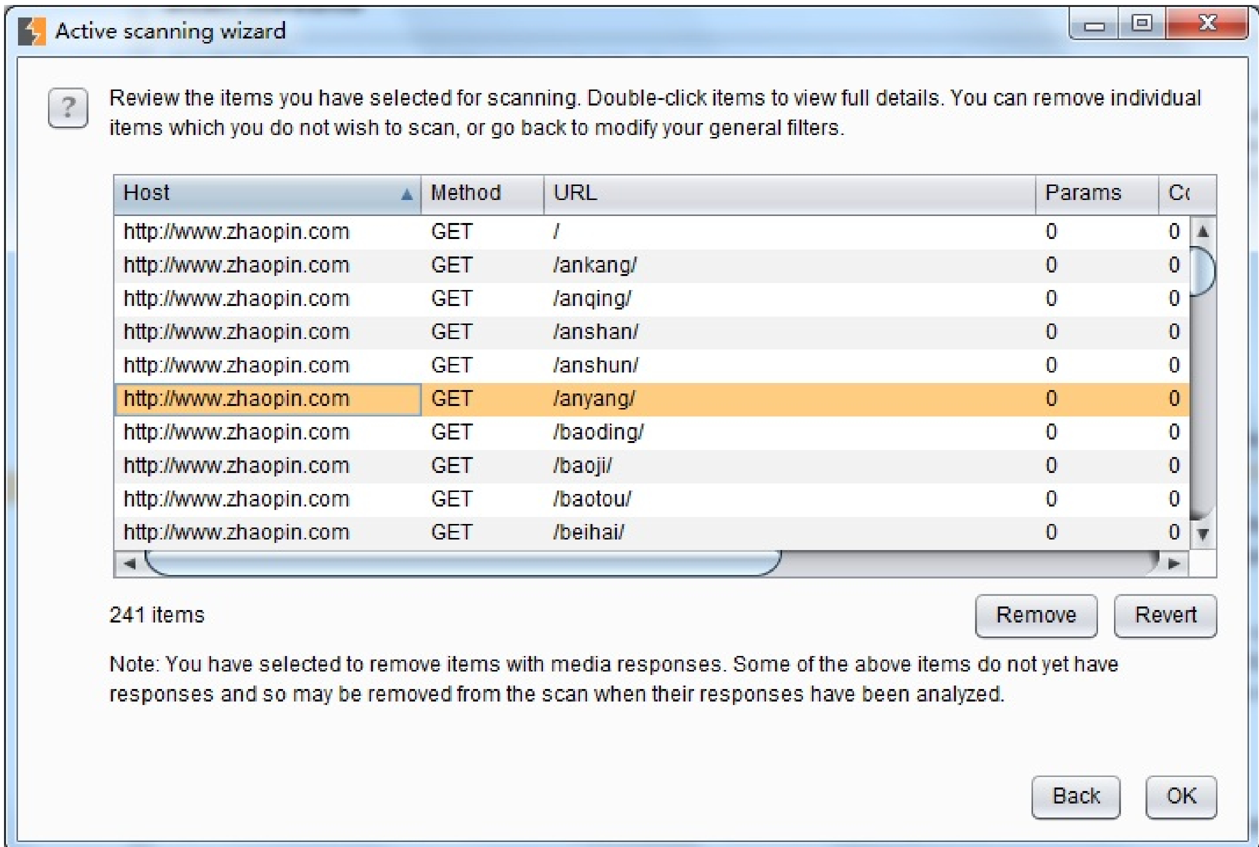


Burp Scanner扫描控制

在对系统做主动扫描时，当我们激活Burp Scanner，扫描控制的相关设置也同时开始了。如下图所示，当我们在Burp Target 的站点地图上的某个URL执行Actively scan this host时，会自动弹出过滤设置。



在这里，我们可以设置扫描时过滤多媒体类型的应答、过滤js、css、图片等静态资源文件。当我们点击【next】按钮，进入扫描路径分支的选择界面。如下图：



以上是Burp Scanner开始扫描前的控制，当我们设置完这些之后，将正式进入扫描阶段。此时，在Scan queue队列界面，会显示扫描的进度、问题总数、请求数和错误统计等信息。

Burp Intruder Repeater Window Help

Target | Proxy | Spider | Scanner | Intruder | Repeater | Sequencer | Decoder | Comparer | Extender | Options | Alerts

Results | Scan queue | Live scanning | Options

#	Host	URL	Status	Issues	Requests	Errors	Insertion points
1	http://www.zhaopin.com	/	25% complete	6	3		3
2	http://www.zhaopin.com	/citymap.html	25% complete	7	11		3
3	http://www.zhaopin.com	/static/analytics.js	25% complete	1	2		3
4	http://www.zhaopin.com	/mobile/mobile.html	25% complete	4	10		3
5	http://www.zhaopin.com	/jobseeker/index_industry.html	25% complete	4	10		3
6	http://www.zhaopin.com	/jobseeker/index_it.html	25% complete		16		3
7	http://www.zhaopin.com	/hydl01/	25% complete	8	3		3
8	http://www.zhaopin.com	/hydl07/	25% complete	8	3		3
9	http://www.zhaopin.com	/hydl03/	25% complete	8	3		3
10	http://www.zhaopin.com	/hydl02/	25% complete	8	2		3
11	http://www.zhaopin.com	/hydl10/	waiting				

在此界面上，我们可以选中某个记录，在右击的弹出菜单中，对扫描进行控制。比如取消扫描、暂停扫描、恢复扫描、转发其他Burp组件等。如下图：

Burp Suite Professional v1.6beta - licensed to LarryLau

Burp Intruder Repeater Window Help

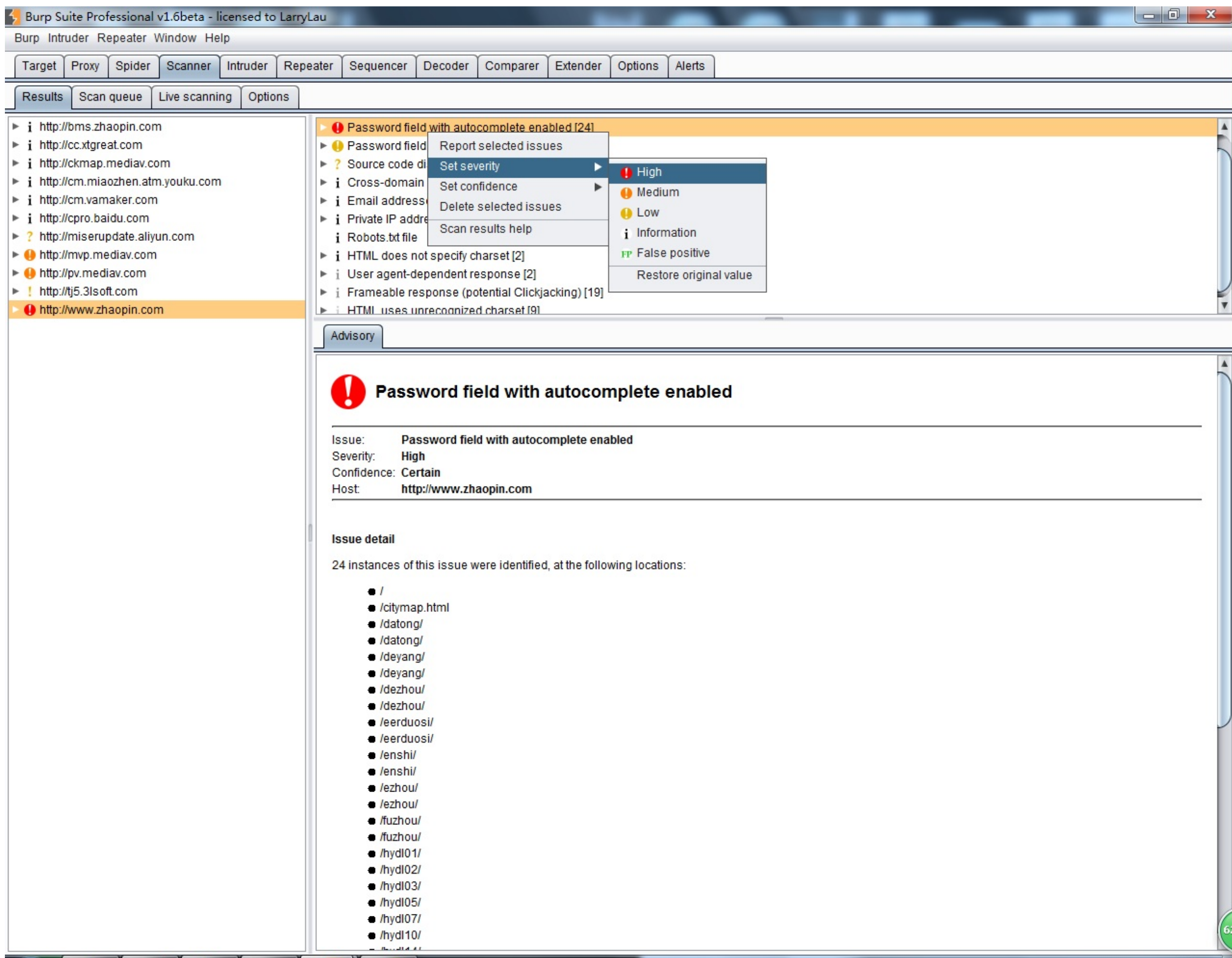
Target | Proxy | Spider | Scanner | Intruder | Repeater | Sequencer | Decoder | Comparer | Extender | Options | Alerts

Results | Scan queue | Live scanning | Options

#	Host	URL	Status	Issues	Requests	Errors	Insertion points
1	http://www.zhaopin.com	/	25% complete	6	16		3
2	http://www.zhaopin.com	/citymap.html	50% complete	7	44		3
3	http://www.zhaopin.com	/static/analytics.js	25% complete	1	23		3
4	http://www.zhaopin.com	/mobile/mobile.html	finished	4	69		3
5	http://www.zhaopin.com	/jobseeker/index_industry.html	75% complete	4	73		3
6	http://www.zhaopin.com	/jobseeker/index_it.html	finished		83		3
7	http://www.zhaopi		50% complete	8	34		3
8	http://www.zhaopi		50% complete	8	42		3
9	http://www.zhaopi		50% complete	8	43		3
10	http://www.zhaopi		50% complete	8	33		3
11	http://www.zhaopi		25% complete	8	11		3
12	http://www.zhaopi		25% complete	8	5		3
13	http://www.zhaopi		waiting				
14	http://www.zhaopi		waiting				
15	http://www.zhaopi		waiting				
16	http://www.zhaopi		waiting				
17	http://www.zhaopin.com	/guangzhou/	waiting				
18	http://www.zhaopin.com	/shenzhen/	waiting				
19	http://www.zhaopin.com	/tianjin/	waiting				
20	http://www.zhaopin.com	/dalian/	waiting				
21	http://www.zhaopin.com	/shenyang/	waiting				
22	http://www.zhaopin.com	/jinan/	waiting				

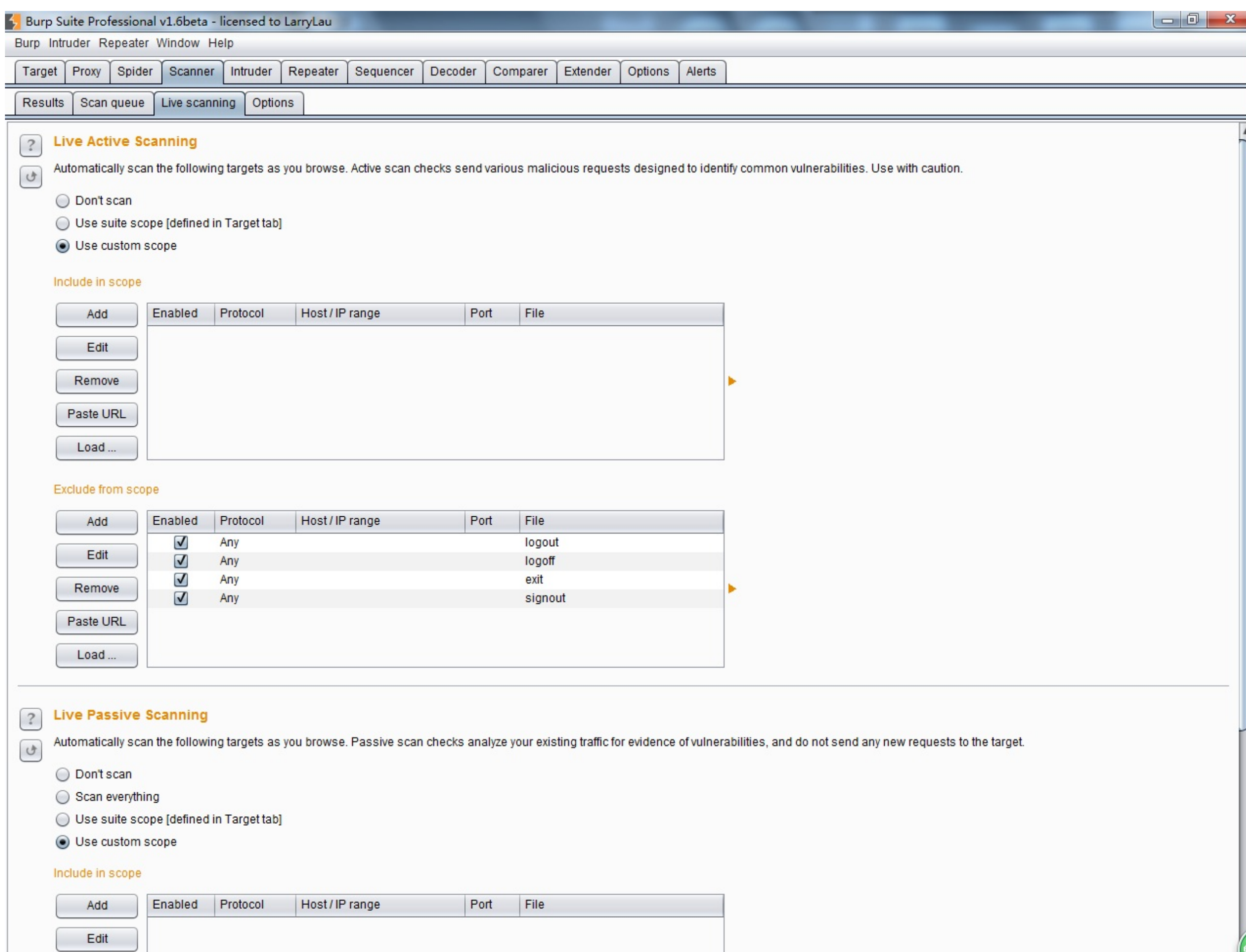
Show details
Scan next
Cancel
Scan again
Delete items
Delete finished items
Automatically delete finished items
Pause scanner
Scan queue help

同时，在Results界面，自动显示队列中已经扫描完成的漏洞明细。



在每一个漏洞的条目上，我们可以选中漏洞。在弹出的右击菜单中，依次选择Set severity，对漏洞的等级进行标识。也可以选择Set confidence，对漏洞是否存在或误报进行标注。

另外，在Live Scanning选项卡中，我们也可以对请求的域、路径、IP地址、端口、文件类型进行控制，如下图：

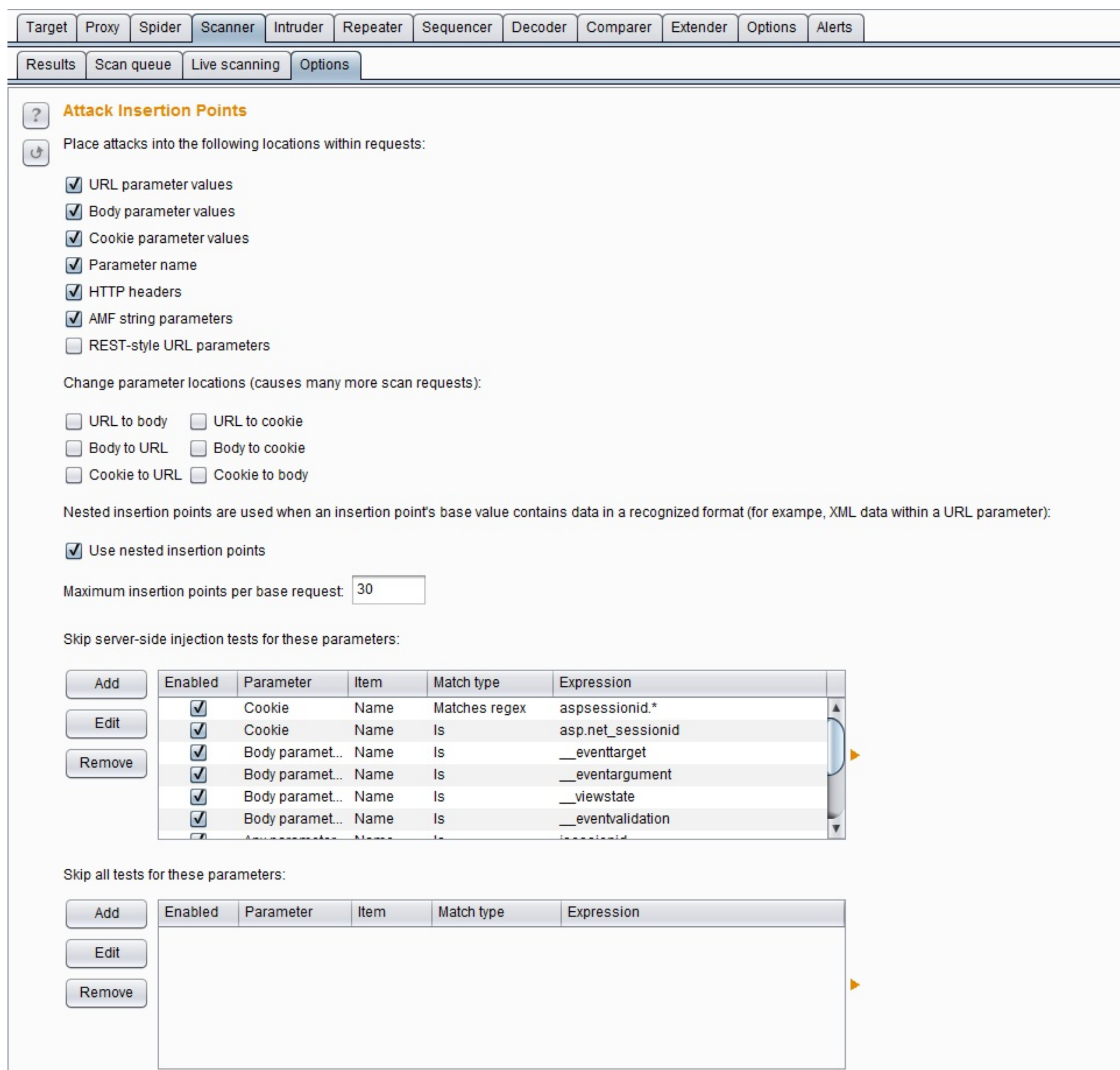


如果你选中了Use suite Scope，则指定条件与你在Burp Target中的Scope配置完全一致，如果你选择了Use customs scope，则可以自己定义Scope，对于Scope的详细配置，请参考Burp Target中的Scope配置相关章节。

Burp Scanner可选项设置

通过前几节的学习，我们已经知道Burp Scanner有主动扫描和被动扫描两个扫描方式，在Options子选项卡中，主要是针对这两种扫描方式在实际扫描中的扫描动作进行设置。具体的设置包含以下部分：

1. 攻击插入点设置（Attack Insertion Points）



Burp Scanner在扫描中，基于原始的请求消息，在每一个插入点构造参数，对原数据进行替换，从而去验证系统漏洞的存在性。通常，以下位置都会被Burp Scanner选择为插入点。

2. URL请求参数
3. Body参数（比如form表单的值，上传文件、XML参数、JSON参数）
4. Cookie参数
5. 参数的名称和个数（通过增加参数个数或者增加参数来验证漏洞）
6. Http Header信息（通过对header信息的篡改来验证漏洞）
7. AFM编码（对flash通信漏洞的验证）
8. REST风格的参数

对于以上的攻击插入点，Burp Scanner还是可以通过改变参数的位置来验证漏洞，Burp Scanner中共有URL to body 、URL to cookie、Body to URL、Body to cookie、Cookie to URL、Cookie to body 六种方式。当我们在扫描验证中，可以根据实际请求，灵活选择位置改变的组合，高效快速地验证漏洞。但我们也应该明白，当我们选中了位置改变来验证漏洞，即选择了Burp发送更多的请求，如果是在生成系统中的测试需要慎重。

另外，Burp的攻击插入点也支持嵌套的方式，这意思是指，如果一个请求的参数值是JSON对象或者XML文本，Burp Scanner在扫描时，可以对JSON对象或XML文本中的属性、属性值进行验证，这会极大地提高了Burp Scanner对漏洞扫描的涉及面。这是由上图中的use nested insertion points的checkbox是否选中去控制的，默认情况下是选中生效的。

当我们设置攻击插入点的同时，我们也可以指定哪些参数进行跳过，不需要进行漏洞验证。在设置时，Burp是按照服务器端参数跳过和所有参数均跳过两种方式来管理的，界面如下图：

Skip server-side injection tests for these parameters:

Add | Edit | Remove

Enabled	Parameter	Item	Match type	Expression
☑	Cookie	Name	Matches regex	aspsessionid.*
☑	Cookie	Name	Is	asp.net_sessionid
☑	Body paramet...	Name	Is	__eventtarget
☑	Body paramet...	Name	Is	__eventargument
☑	Body paramet...	Name	Is	__viewstate
☑	Body paramet...	Name	Is	__eventvalidation

Skip all tests for these parameters:

Add | Edit | Remove

Enabled	Parameter	Item	Match type	Expression

2 主动扫描引擎设置（Active Scanning Engine）

Active Scanning Engine

These settings control the engine used for making HTTP requests when doing active scanning.

Number of threads: 10

Number of retries on network failure: 3

Pause before retry (milliseconds): 2000

☐ Throttle between requests (milliseconds): 500

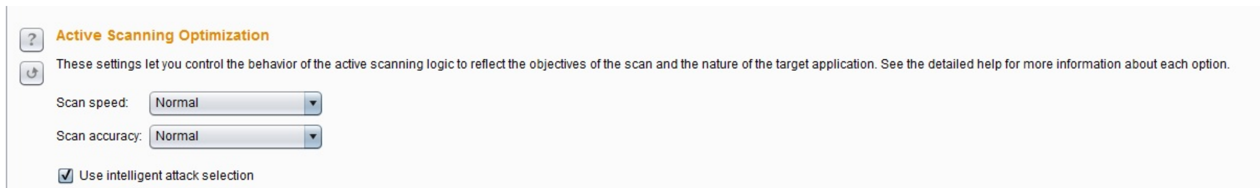
☐ Add random variations to throttle

☑ Follow redirections where necessary

主动扫描引擎设置主要是用来控制主动扫描时的线程并发数、网络失败重试间隔、网络失败重试次数、请求延迟、是否跟踪重定向。其中请求延迟设置（Throttle between requests）和其子选项延迟随机数 （Add random variations to throttle）在减少应用负荷，模拟人工测试，使得扫描更加隐蔽，而不易被网络安全设备检测出来。 至于这些参数的具体设置，需要你根据服务器主机的性能、网络带宽、客户端测试机的性能做相应的调整。一般来说，如果您发

现该扫描运行缓慢，但应用程序表现良好，你自己的CPU利用率较低，可以增加线程数，使您的扫描进行得更快。如果您发现发生连接错误，应用程序正在放缓，或你自己的电脑很卡，你应该减少线程数，加大对网络故障的重试次数和重试之间的间隔。

3.主动扫描优化设置（Active Scanning Optimization）



此选项的设置主要是为了优化扫描的速度和准确率，尽量地提高扫描速度的同时降低漏洞的误报率。 扫描速度（Scan speed）分快速、普通、彻底三个选项，不同的选项对应于不同的扫描策略，当选择彻底扫描（Thorough）时，Burp会发送更多的请求，对漏洞的衍生类型会做更多的推导和验证。而当你选择快速扫描（Fast），Burp则只会做一般性的、简单的漏洞验证。 扫描精准度（Scan accuracy）也同样分为三个选项：最小化假阴性（Minimize false negatives）、普通、最小化假阳性（Minimize false positives）。扫描精准度主要是用来控制Burp的扫描过程中针对漏洞的测试次数。当我们选择最小化假阳性时，Burp会做更多的验证测试，来防止假阳性漏洞的存在，但也是恰恰基于此，当Burp做更多的验证测试时，可能存在恰好无法获取应答的误报，增加了漏洞的噪音。 智能攻击选择（Use intelligent attack selection ）这个选项通过智能地忽略一些攻击插入点基值的检查，比如说一个参数值包含不正常出现在文件名中的字符，Burp将跳过文件路径遍历检查此参数，使用此选项可加速扫描，并降低在提升扫描速度的同时会导致漏报率上升的风险。

4.主动扫描范围设置（Active Scanning Areas）

Active Scanning Areas

These settings control the types of checks performed during active scanning.

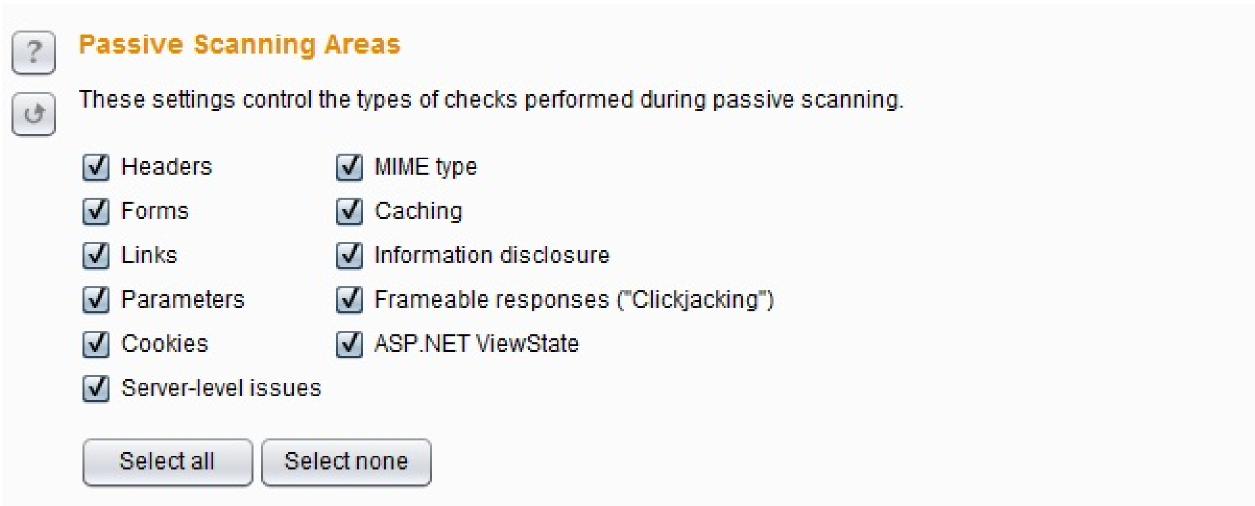
- [x] SQL injection
  - [x] Error-based
  - [x] Time-delay tests
  - [x] Boolean condition tests
  - [x] MSSQL-specific tests
  - [x] Oracle-specific tests
  - [x] MySQL-specific tests
- [x] OS command injection
  - [x] Informed
  - [x] Blind
- [x] Reflected XSS
- [x] Stored XSS
- [x] File path traversal
- [x] Remote file inclusion
- [x] HTTP header injection
- [x] XML / SOAP injection
- [x] LDAP injection
- [x] Open redirection
- [x] Header manipulation
- [x] Server-level issues

Select all　Select none

在主动扫描过程中，你可以根据你的扫描时间、关注的重点、可能性存在的漏洞类型等情况，选择不同的扫描范围。这里可选择的扫描范围有：

- SQL注入 -可以使不同的测试技术（基于误差，时间延迟测试和布尔条件测试），并且也使检查所特有的单独的数据库类型（MSSQL，Oracle和MySQL的）。
- 操作系统命令注入 - （信息通知和盲注）。
- 反射式跨站点脚本
- 存储的跨站点脚本
- 文件路径遍历
- HTTP头注入
- XML/ SOAP注入
- LDAP注入
- URL重定向
- http消息头可操纵
- 服务器的问题

5.被动扫描范围设置（Passive Scanning Areas）



因为被动扫描不会发送新的请求，只会对原有数据进行分析，其扫描范围主要是请求和应答消息中的如下参数或漏洞类型：Headers、Forms、Links、Parameters、Cookies、MIME type、Caching、敏感信息泄露、Frame框架点击劫持、ASP.NET ViewState 。

第八章 如何使用Burp Intruder

Burp Intruder作为Burp Suite中一款功能极其强大的自动化测试工具，通常被系统安全渗透测试人员被使用在各种任务测试的场景中。本章我们主要学习的内容有：

- Intruder使用场景和操作步骤
- Payload类型与处理
- Payload 位置和攻击类型
- 可选项设置（Options）
- Intruder 攻击和结果分析

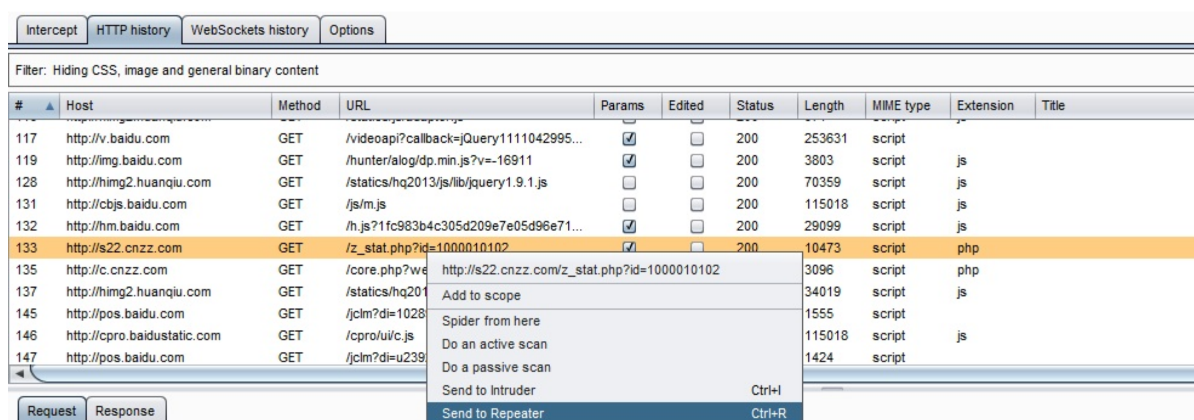
Intruder使用场景和操作步骤

在渗透测试过程中，我们经常使用Burp Intruder，它的工作原理是：Intruder在原始请求数据的基础上，通过修改各种请求参数，以获取不同的请求应答。每一次请求中，Intruder通常会携带一个或多个有效攻击载荷（Payload),在不同的位置进行攻击重放，通过应答数据的比对分析来获得需要的特征数据。Burp Intruder通常被使用在以下场景：

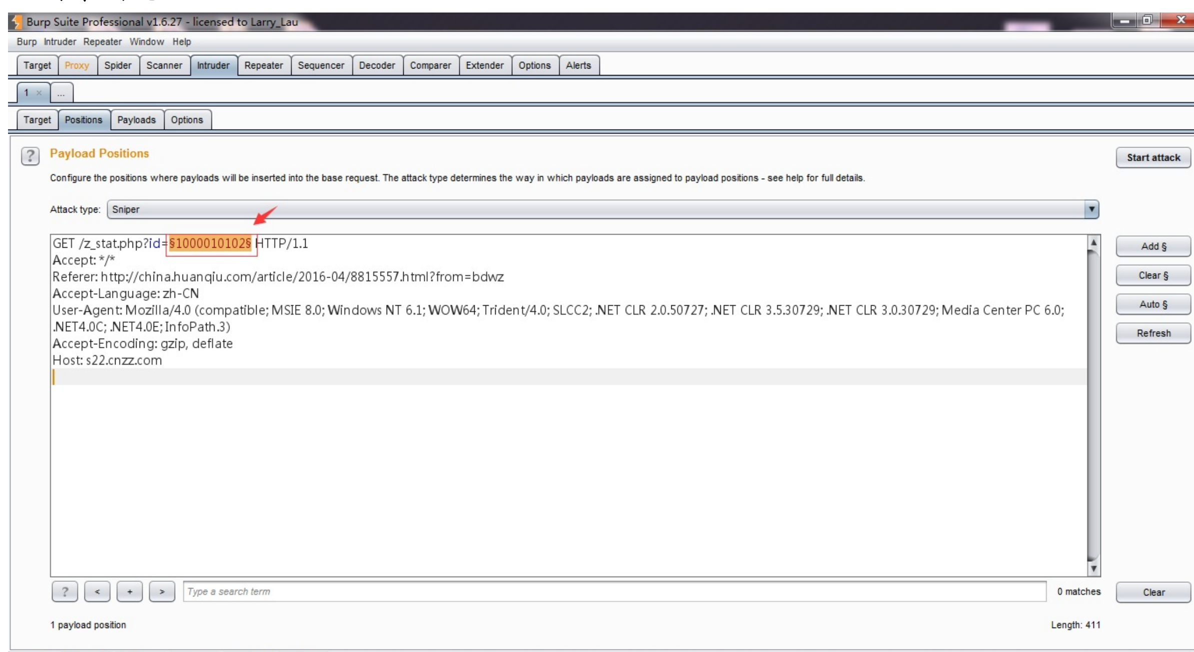
1. 标识符枚举 Web应用程序经常使用标识符来引用用户、账户、资产等数据信息。例如，用户名，文件ID和账户号码。
2. 提取有用的数据 在某些场景下，而不是简单地识别有效标识符，你需要通过简单标识符提取一些其他的数据。比如说，你想通过用户的个人空间id，获取所有用户在个人空间标准的昵称和年龄。
3. 模糊测试 很多输入型的漏洞，如SQL注入，跨站点脚本和文件路径遍历可以通过请求参数提交各种测试字符串，并分析错误消息和其他异常情况，来对应用程序进行检测。由于的应用程序的大小和复杂性，手动执行这个测试是一个耗时且繁琐的过程。这样的场景，您可以设置Payload，通过Burp Intruder自动化地对Web应用程序进行模糊测试。

通常来说，使用Burp Intruder进行测试，主要遵循以下步骤：

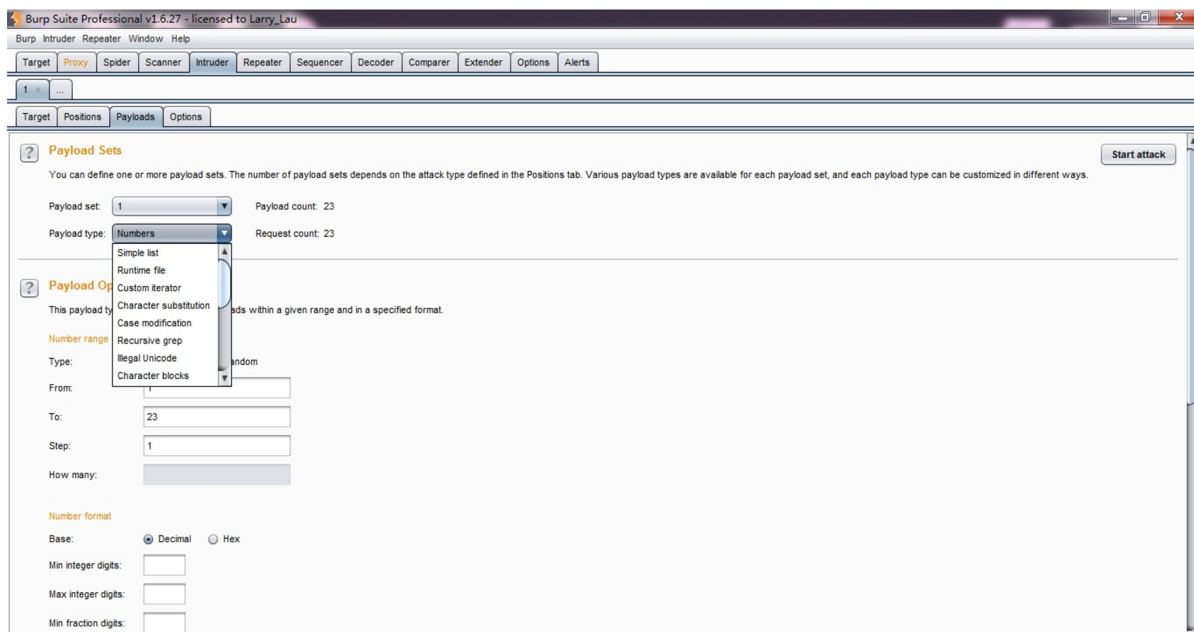
1. 确认Burp Suite安装正确并正常启动，且完成了浏览器的代理设置。
2. 进入Burp Proxy选项卡，关闭代理拦截功能。
3. 进行历史日志（History）子选项卡，查找可能存在问题的请求日志，并通过右击菜单，发送到Intruder。



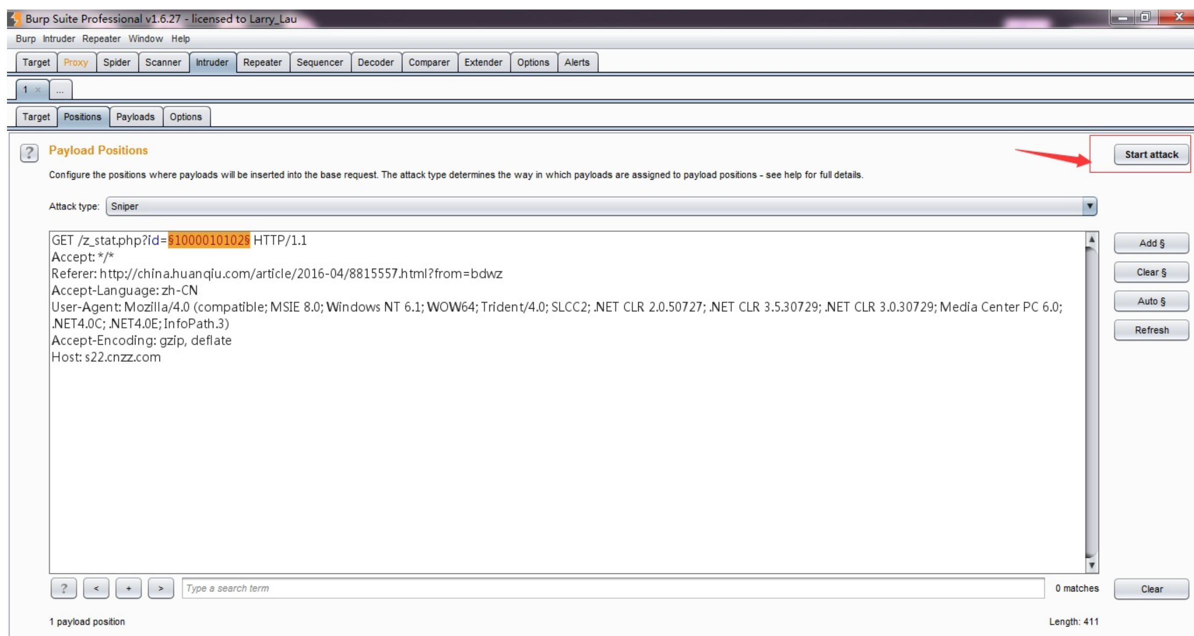
4. 进行Intruder 选项卡，打开Target和Positions子选项卡。这时，你会看到上一步发送过来的请求消息。



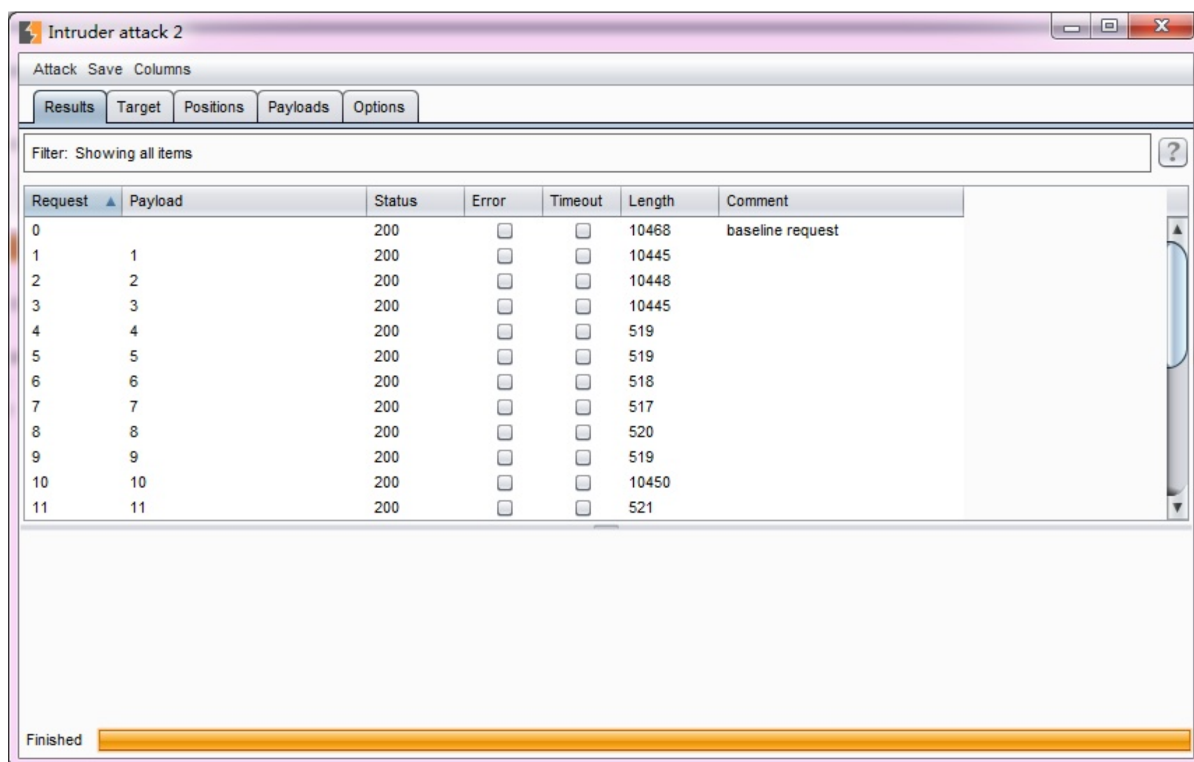
5. 因为我们了解到Burp Intruder攻击的基础是围绕刚刚发送过来的原始请求信息，在原始信息指定的位置上设置一定数量的攻击载荷Payload，通过Payload来发送请求获取应答消息。默认情况下，Burp Intruder会对请求参数和Cookie参数设置成Payload position，前缀添加 $符合，如上图红色标注位置所示。当发送请求时，会将$标识的参数替换为Payload。
6. 在Position界面的右边，有【Add $】、【Clear $】、【Auto $】、【Refersh $】四个按钮，是用来控制请求消息中的参数在发送过程中是否被Payload替换，如果不想被替换，则选择此参数，点击【Clear $】,即将参数前缀$去掉。
7. 当我们打开Payload 子选项卡，选择Payload的生成或者选择策略，默认情况下选择“Simple list”, 当然你也可以通过下拉选择其他Payload类型或者手工添加。



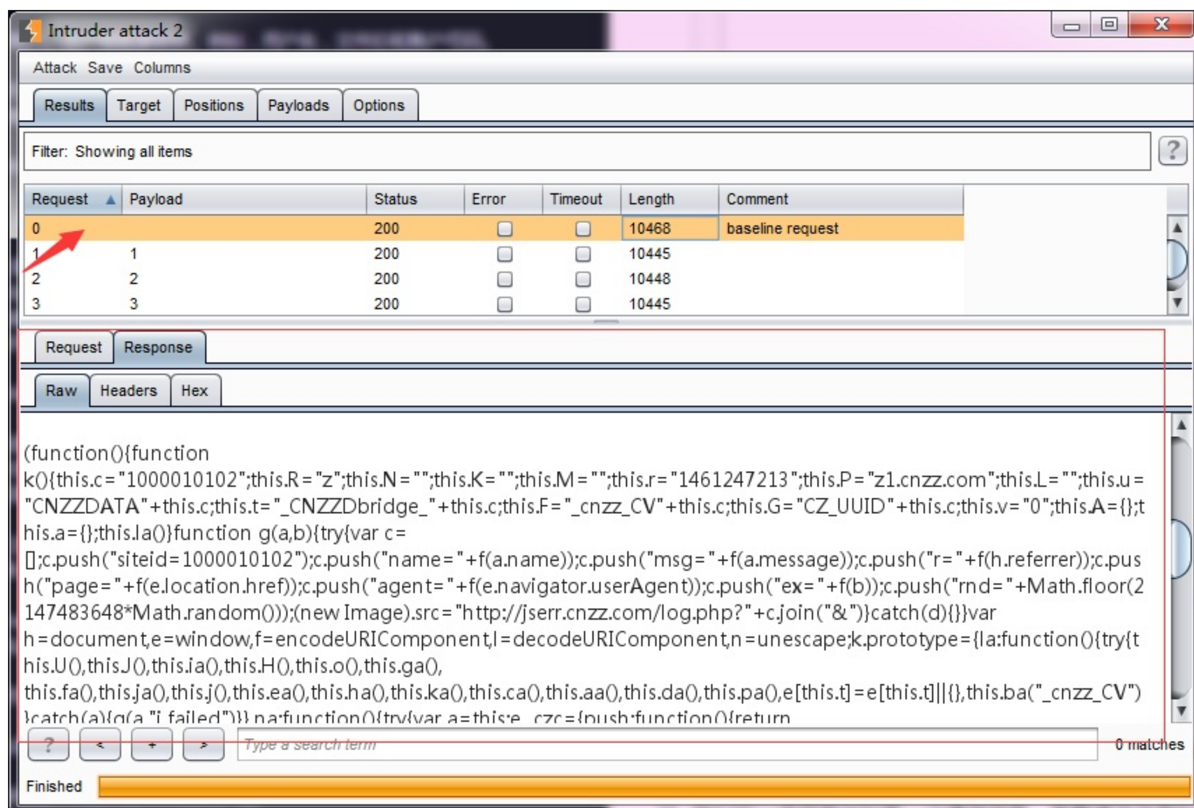
8. 此时，我们再回到Position界面，在界面的右上角，点击【Start attack】，发起攻击。



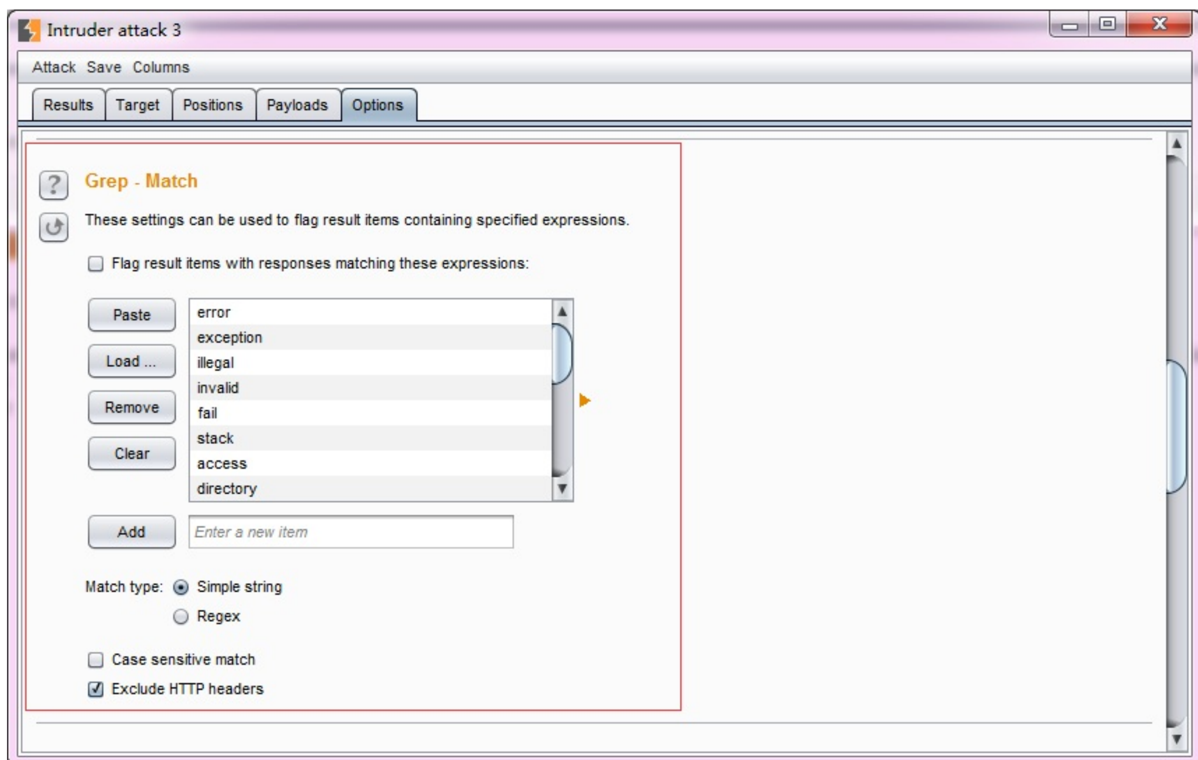
9. 此时，Burp会自动打开一个新的界面，包含攻击执行的情况、Http状态码、长度等结果信息。



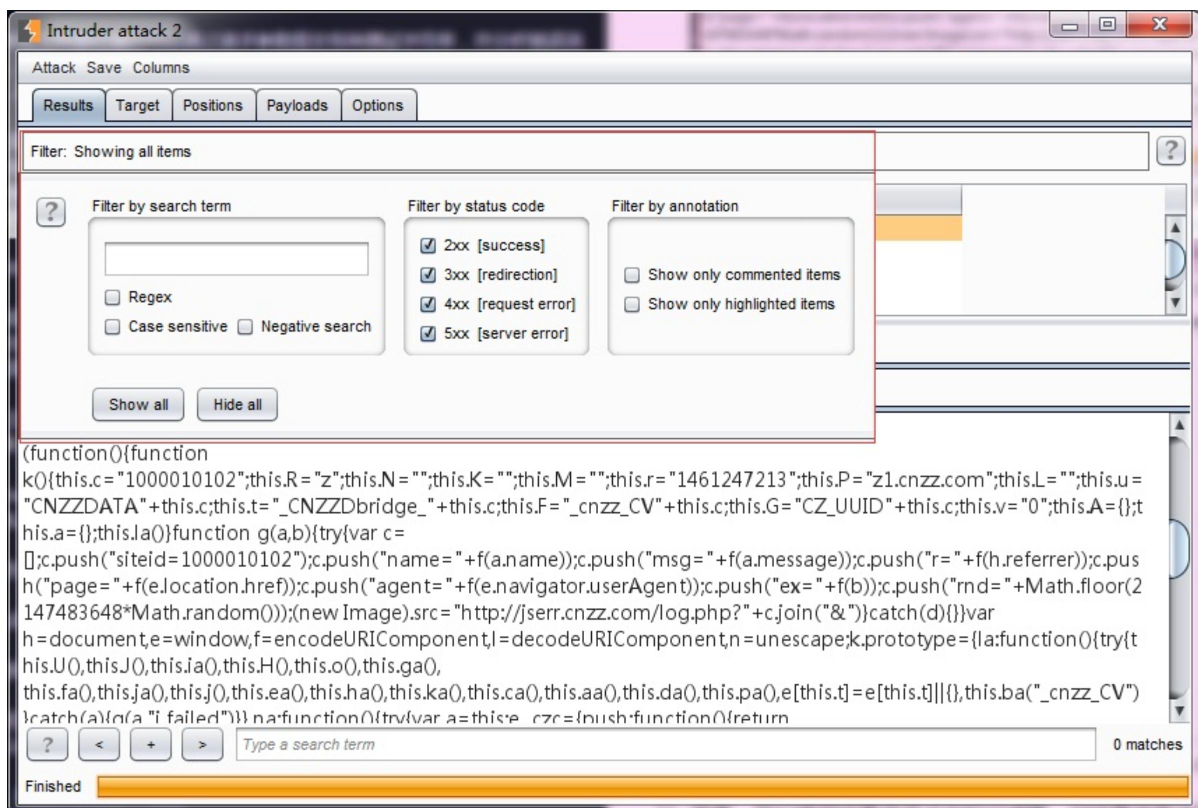
10. 我们可以选择其中的某一次通信信息，查看请求消息和应答消息的详细。



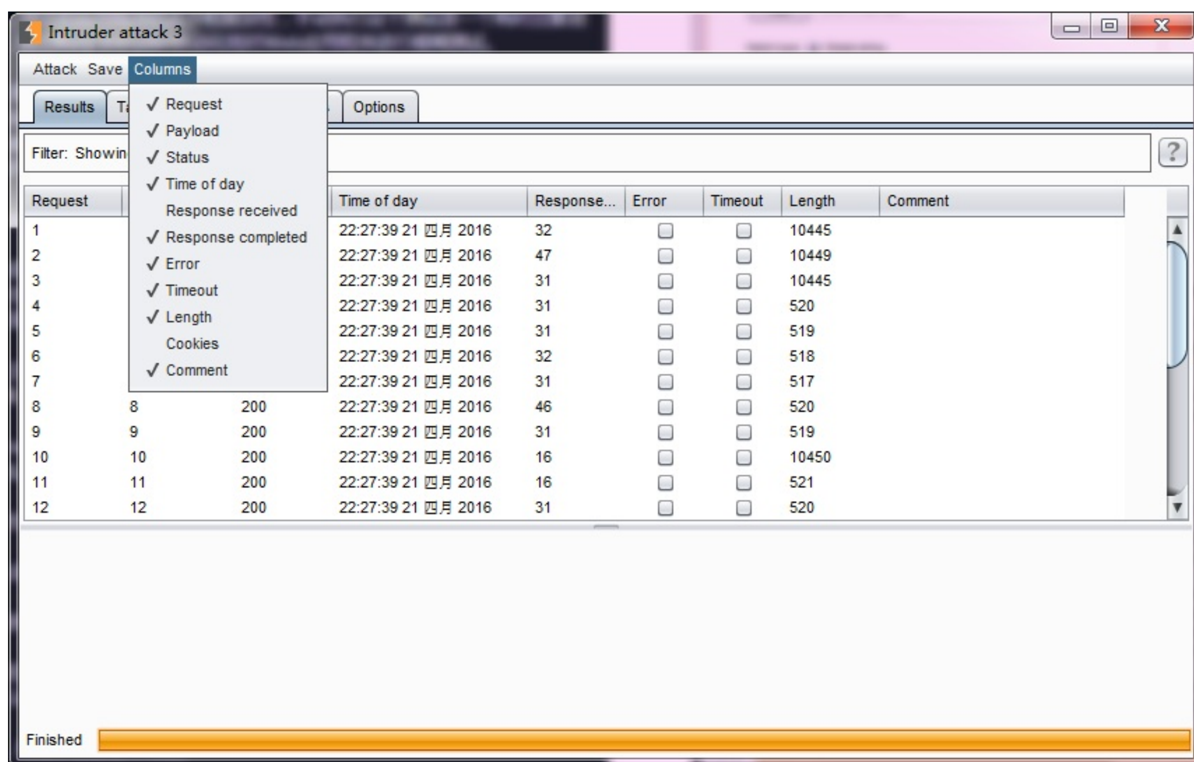
11. 很多时候，为了更好的标明应答消息中是否包含有我们需要的信息，通常在进行攻击前，会进行Options选项的相关配置，使用最多的为正则表达式匹配（Grep - Match）。



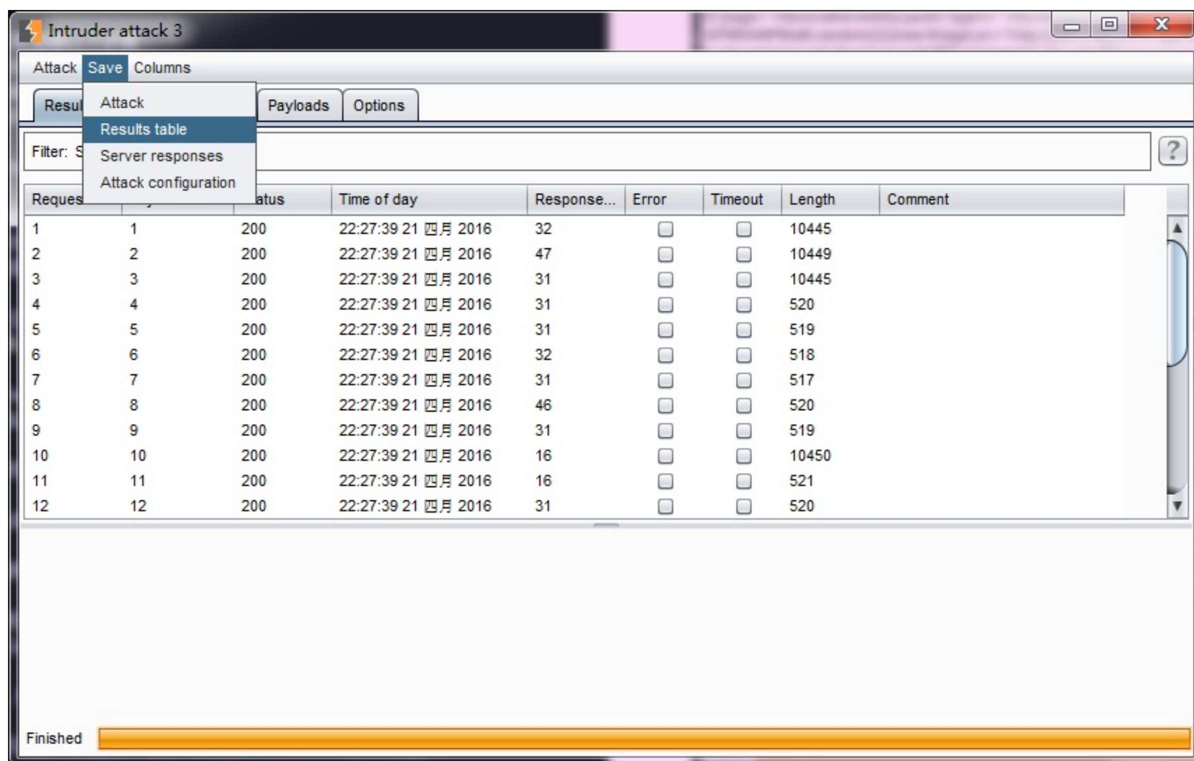
12. 或者，我们使用结果选项卡中的过滤器，对结果信息进行筛选。



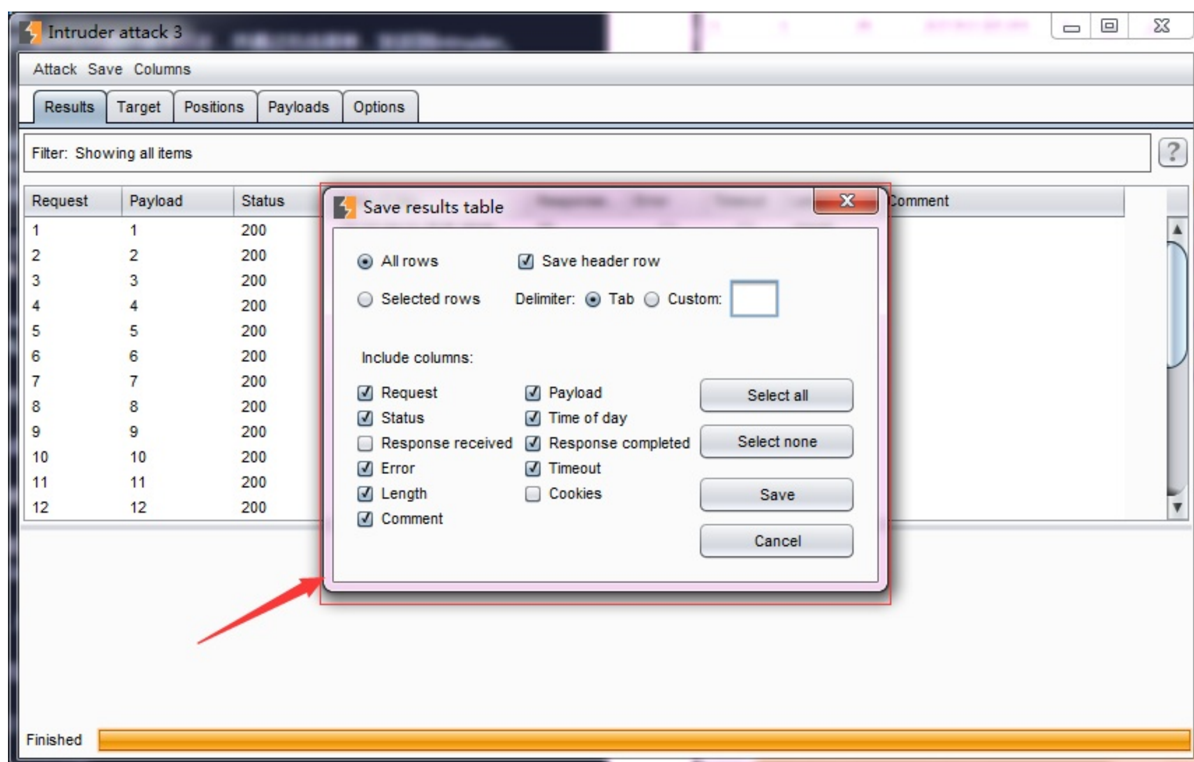
13. 同时，结果选项卡中所展示的列我们是可以进行指定的，我们可以在菜单Columns进行设置。



14. 最后，选择我们需要的列，点击【Save】按钮，对攻击结果进行保存。



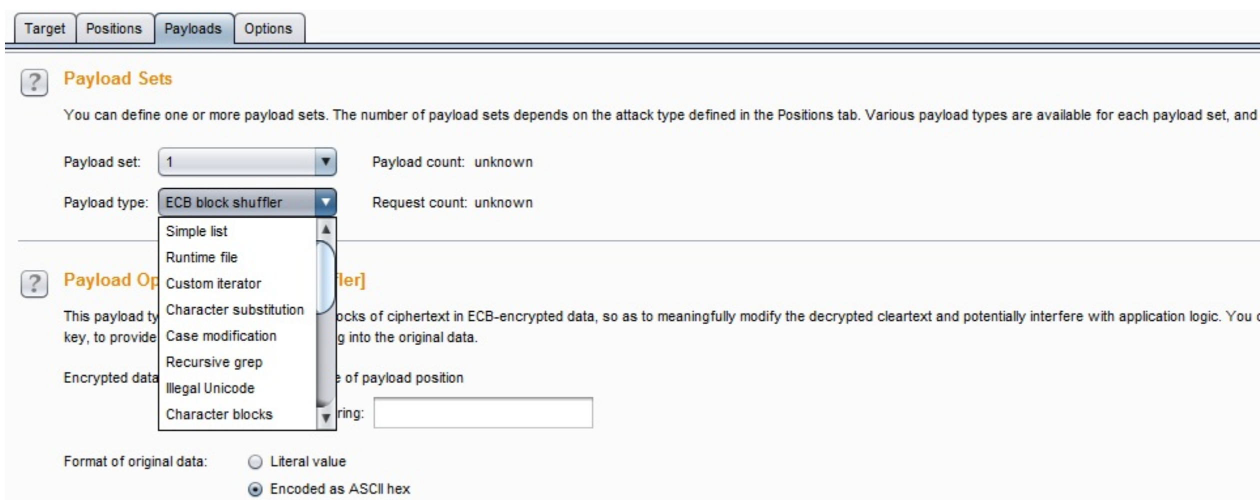
15. 当然，保存之前我们也可以对保存的内容进行设置。



以上这些，是Burp Intruder一次完成的操作步骤，在实际使用中，根据每一个人的使用习惯，会存在或多或少的变动。而每一个环节中涉及的更详细的配置，将在接下来的章节中做更细致的阐述。

Payload类型与处理

在Burp Intruder的Payload选项卡中，有Payload集合的设置选项，包含了经常使用的Payload类型，共18种。



他们分别是：

- 简单列表（Simple list）——最简单的Payload类型，通过配置一个字符串列表作为Payload，也可以手工添加字符串列表或从文件加载字符串列表。其设置界面如下图

Target | Positions | Payloads | Options

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack

Payload set: 1 Payload count: 0

Payload type: Simple list Request count: 0

Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

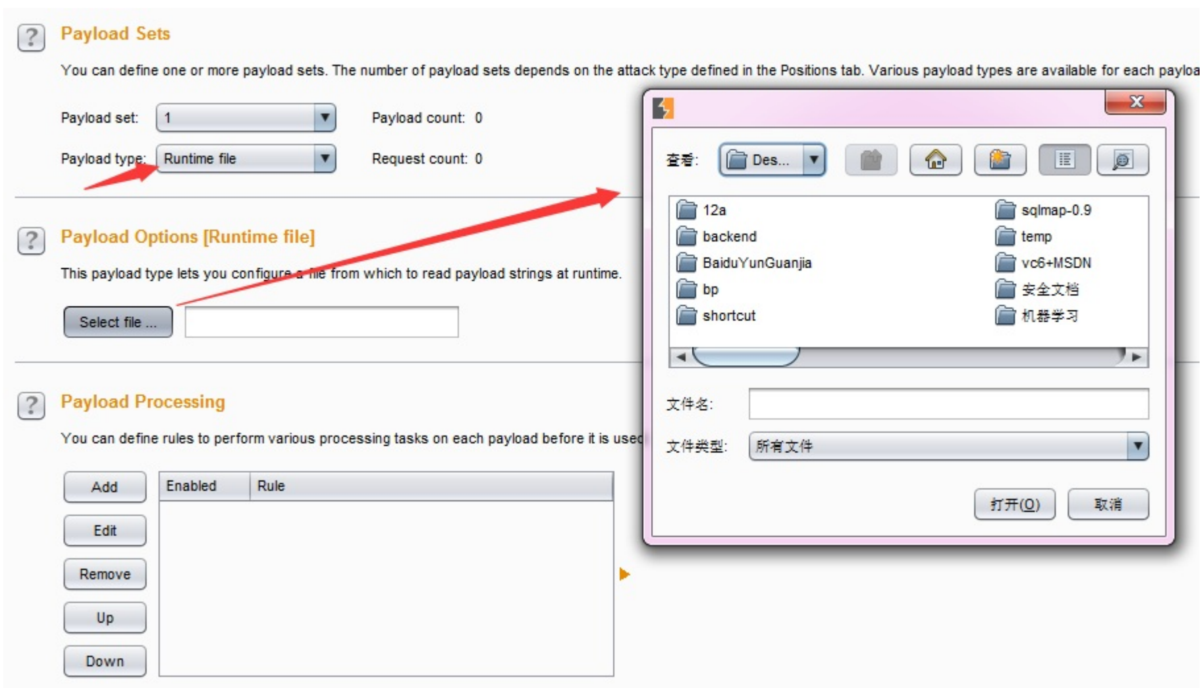
Paste
Load ...
Remove
Clear
Add
Enter a new item
Add from list ...

手工添加或文件加载

选择Payload列表

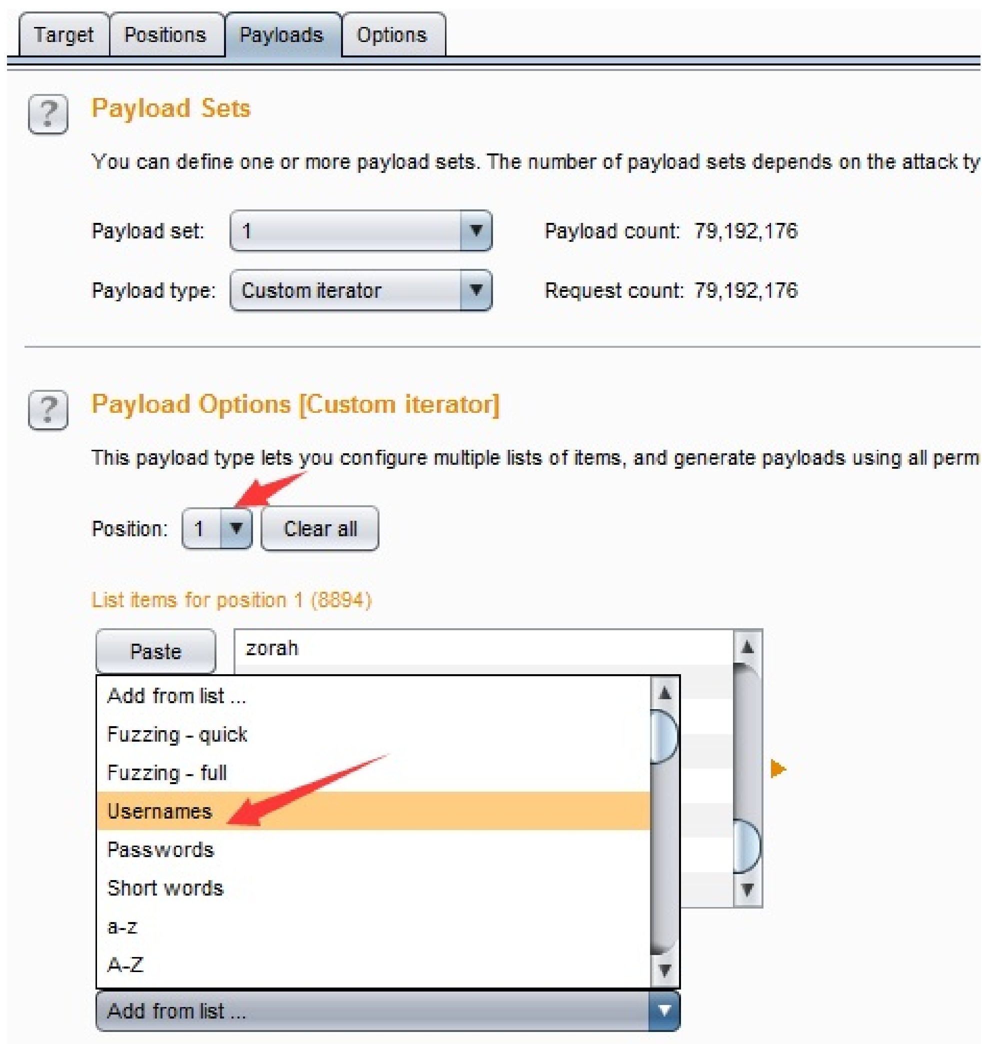
在此操作界面上，选择的Payload列表中，已经预定义了一组简单Payload列表，包括XSS脚本、CGI脚本、SQL注入脚本、数字、大写字母、小写字母、用户名、密码、表单域的字段名、IIS文件名和目录名等等，极大地方便了渗透测试人员的使用。

- 运行时文件（Runtime file）——指定文件，作为相对应Payload位置上的Payload列表。其设置界面如下图：

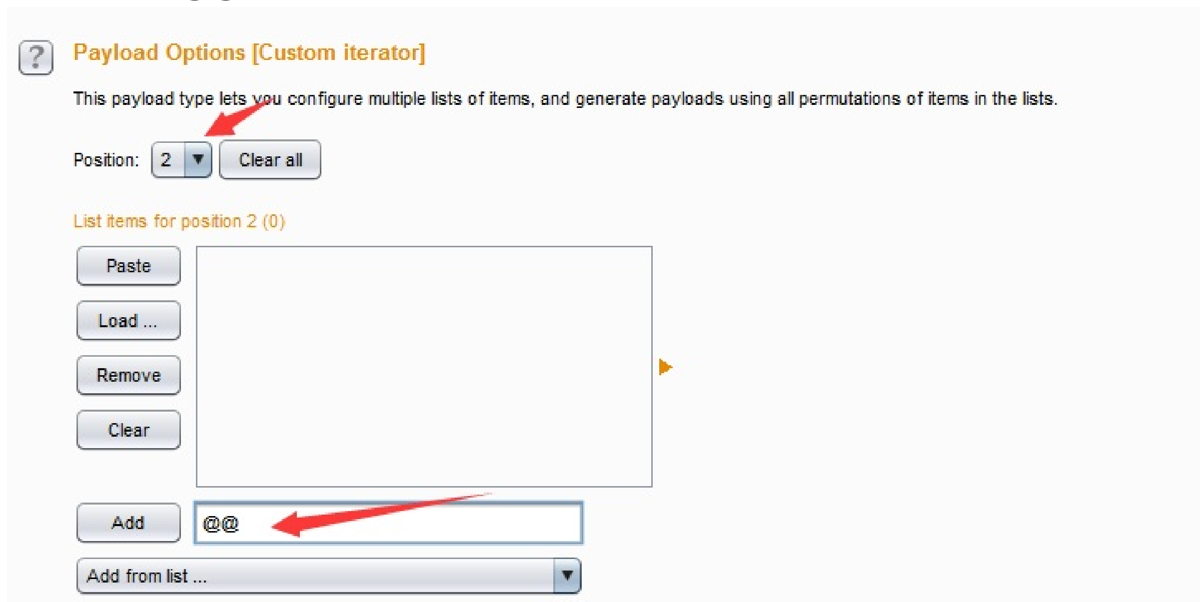


当我们如上图所示，指定Payload set的位置1使用的Payload类型为Runtime file时，下方的Payload Options将自动改变为文件选择按钮和输入框，当我们点击【select file】选择文件时，将弹出图中所示的对话框，选择指定的Payload文件。运行时，Burp Intruder将读取文件的每一行作为一个Payload。

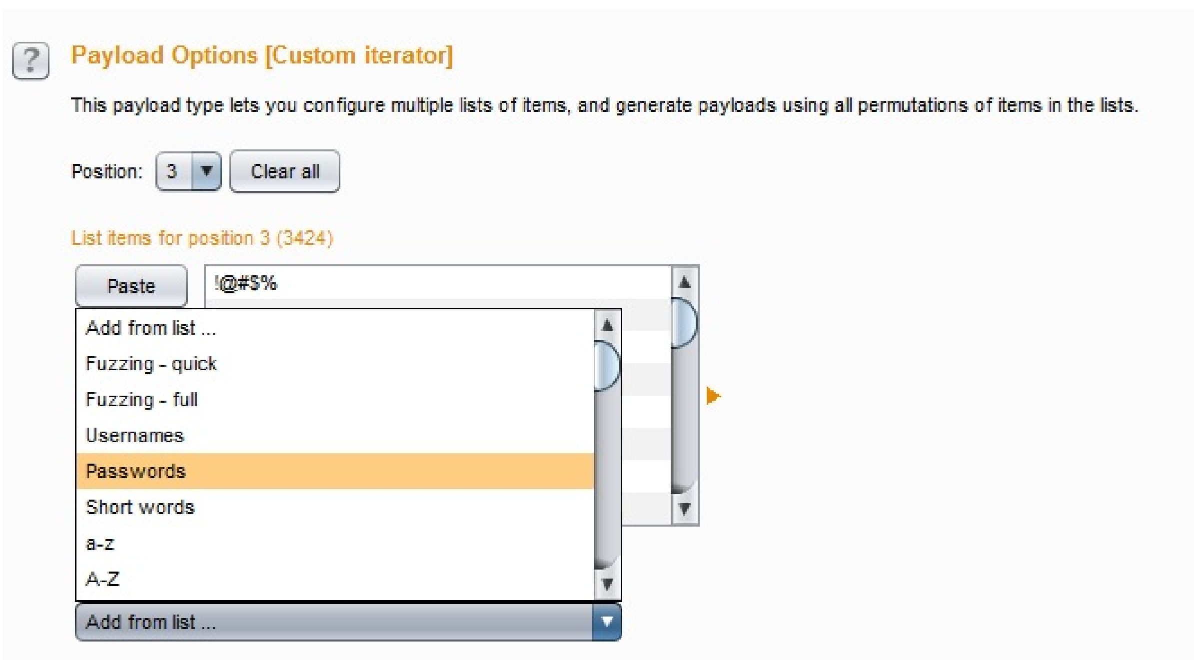
- 自定义迭代器（Custom iterator）——这是一款功能强大的Payload，它共有8个占位，每一个占位可以指定简单列表的Payload类型，然后根据占位的多少，与每一个简单列表的Payload进行笛卡尔积，生成最终的Payload列表。例如，某个参数的值格式是username@@password，则设置此Payload的步骤是：位置1，选择Usernames



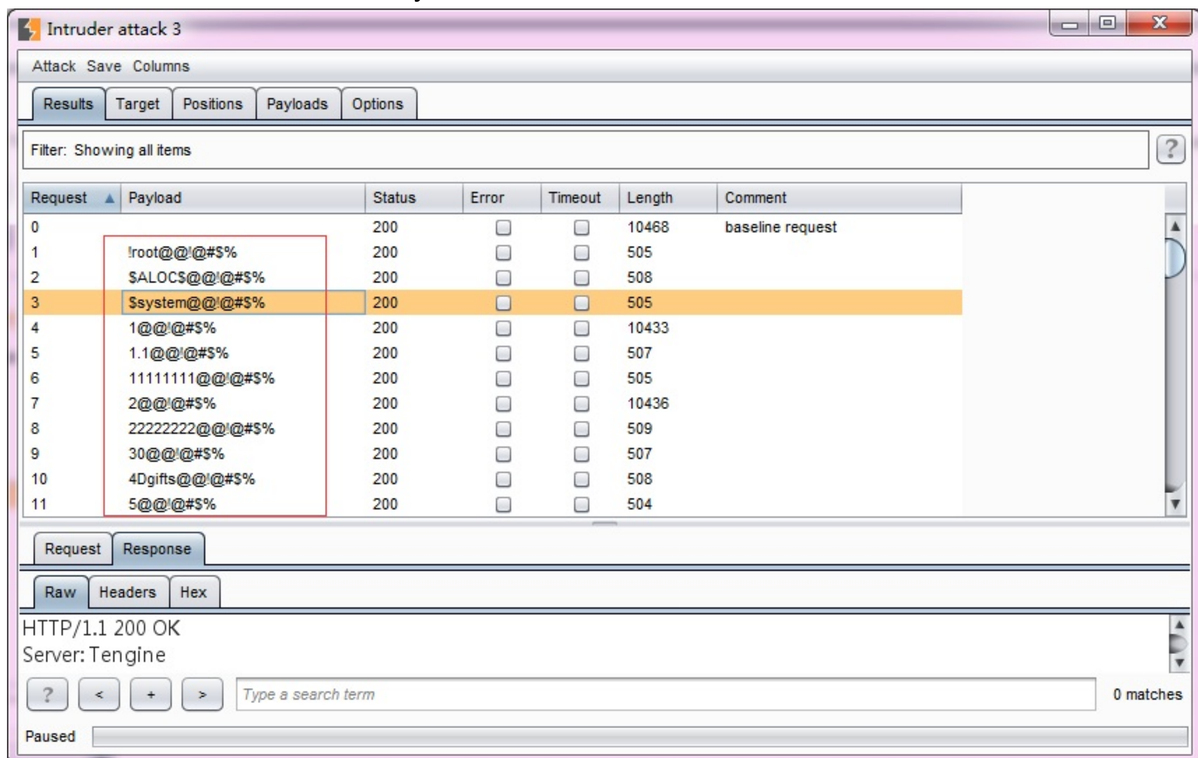
接着，指定位置2，输入值@@



最后指定位置3，选择Passwords



当我们开始攻击时，生成的Payload值如图所示



- 字符串替换（Character substitution）——顾名思义，此种Payload的类型是对预定义的字符串进行替换后生成新的Payload。比如说，预定义字符串为ABCD，按照下图的替换规则设置后，将对AB的值进行枚举后生成新的Payload。

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the

Payload set: 1　　Payload count: 8 (approx)

Payload type: Character substitution　　Request count: 8 (approx)

Payload Options [Character substitution]

This payload type lets you configure a list of strings and apply various character substitutions to each item.

Character substitutions　　替换规则

a > 4	b > 8	e > 3	g > 6
i > 1	o > 0	s > 5	t > 7
z > 2	>	>	>
>	>	>	>

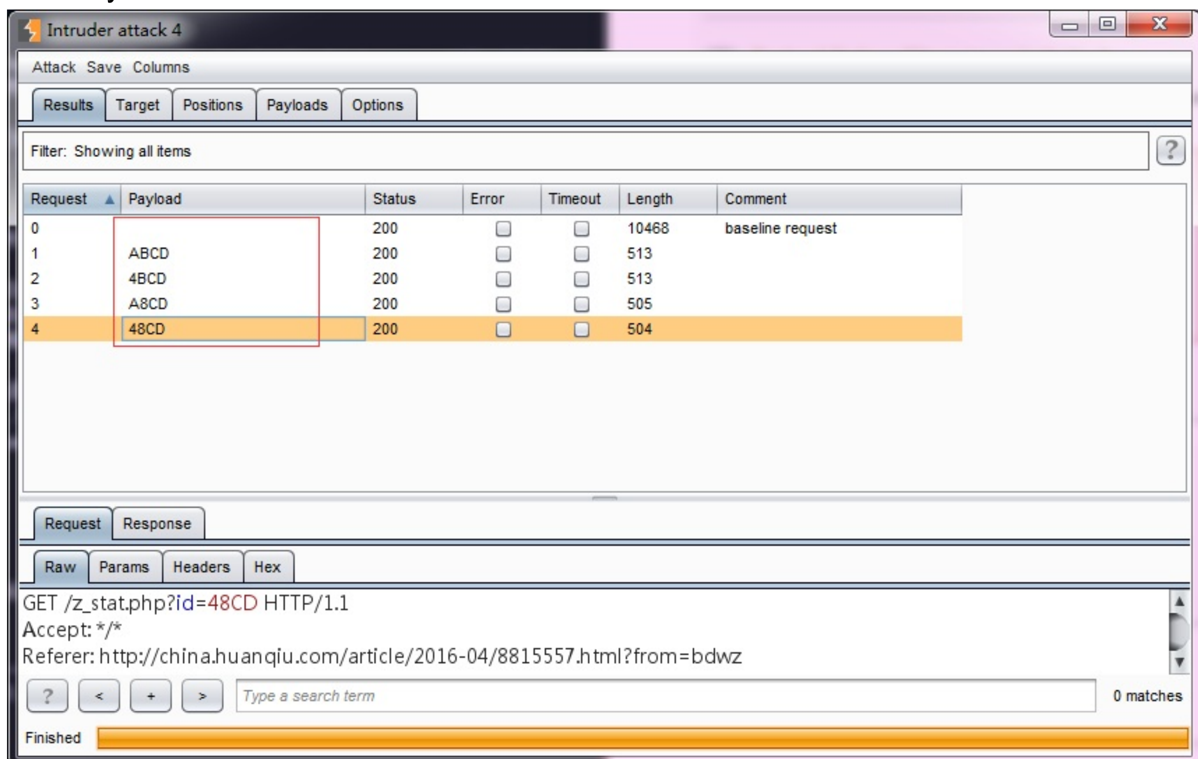
Case sensitive match

Items (1)

Paste　Load ...　Remove　Clear

ABCD

生成的Payload如下图所示，分别替换了上图中的a和b的值为4与8



- 大小写替换（Case modification）——对预定义的字符串，按照大小写规则，进行替换。比如说，预定义的字符串为Peter Wiener，则按照下图的设置后，会生成新的Payload。

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions ta

Payload set: 1　　Payload count: 3 (approx)

Payload type: Case modification　　Request count: 3 (approx)

Payload Options [Case modification]

This payload type lets you configure a list of strings and apply various case modifications to each item.

Case modifications

- [x] No change
- [x] To lower case
- [x] To upper case
- [x] To Propername
- [x] To ProperName

Items (1)

Paste　Load ...　Remove　Clear

Peter Wiener

生成的Payload如下

Intruder attack 13

Attack　Save　Columns

Results　Target　Positions　Payloads　Options

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
1	Peter Wiener	200	☐	☐	530	
2	Peter wiener	200	☐	☐	514	
3	PETER WIENER	200	☐	☐	516	
4	peter wiener	200	☐	☐	514	

Request　Response

Raw　Headers　Hex

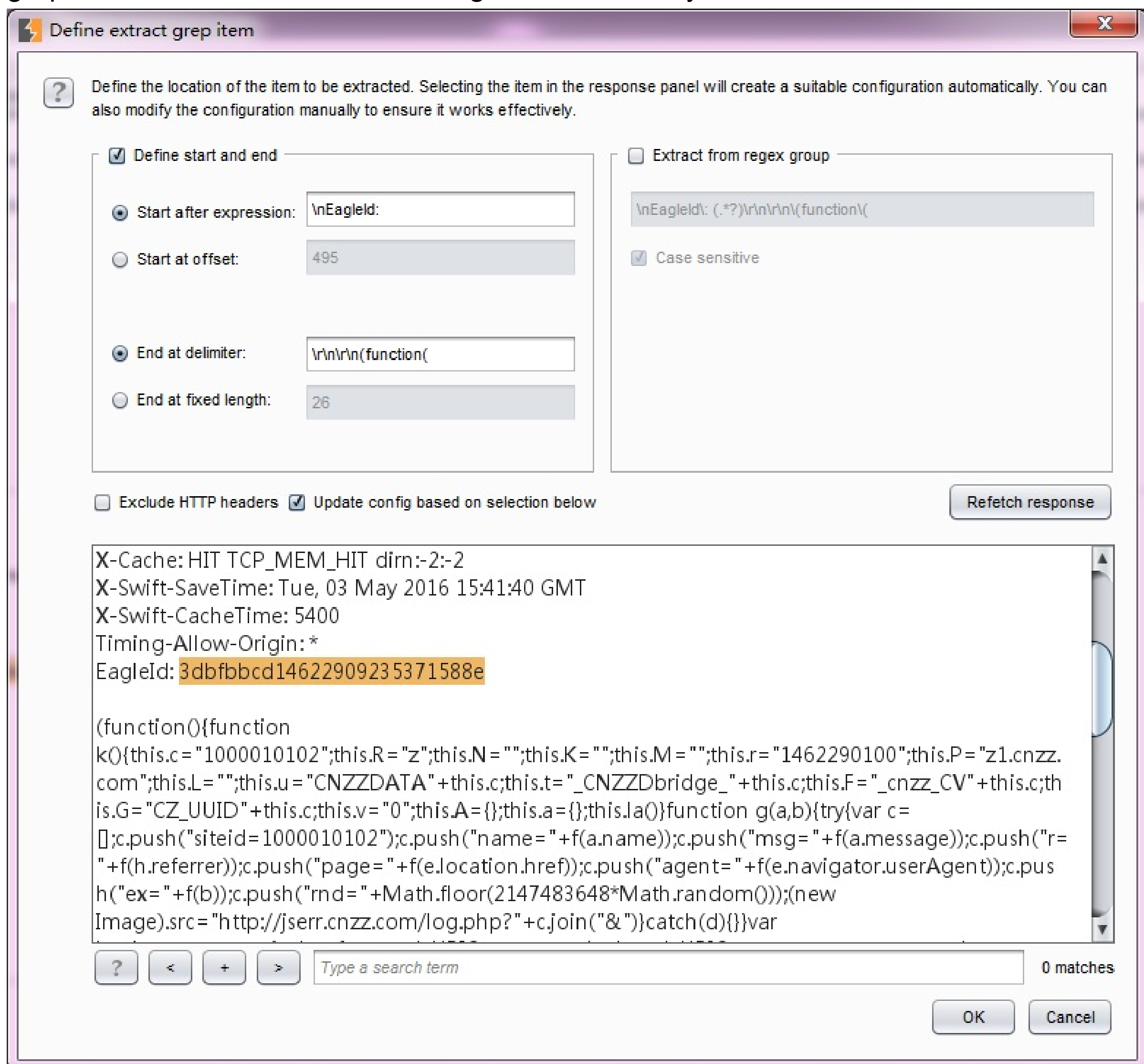
```
HTTP/1.1 200 OK
Server: Tengine
Content-Type: application/javascript
Content-Length: 9941
```

Type a search term　0 matches

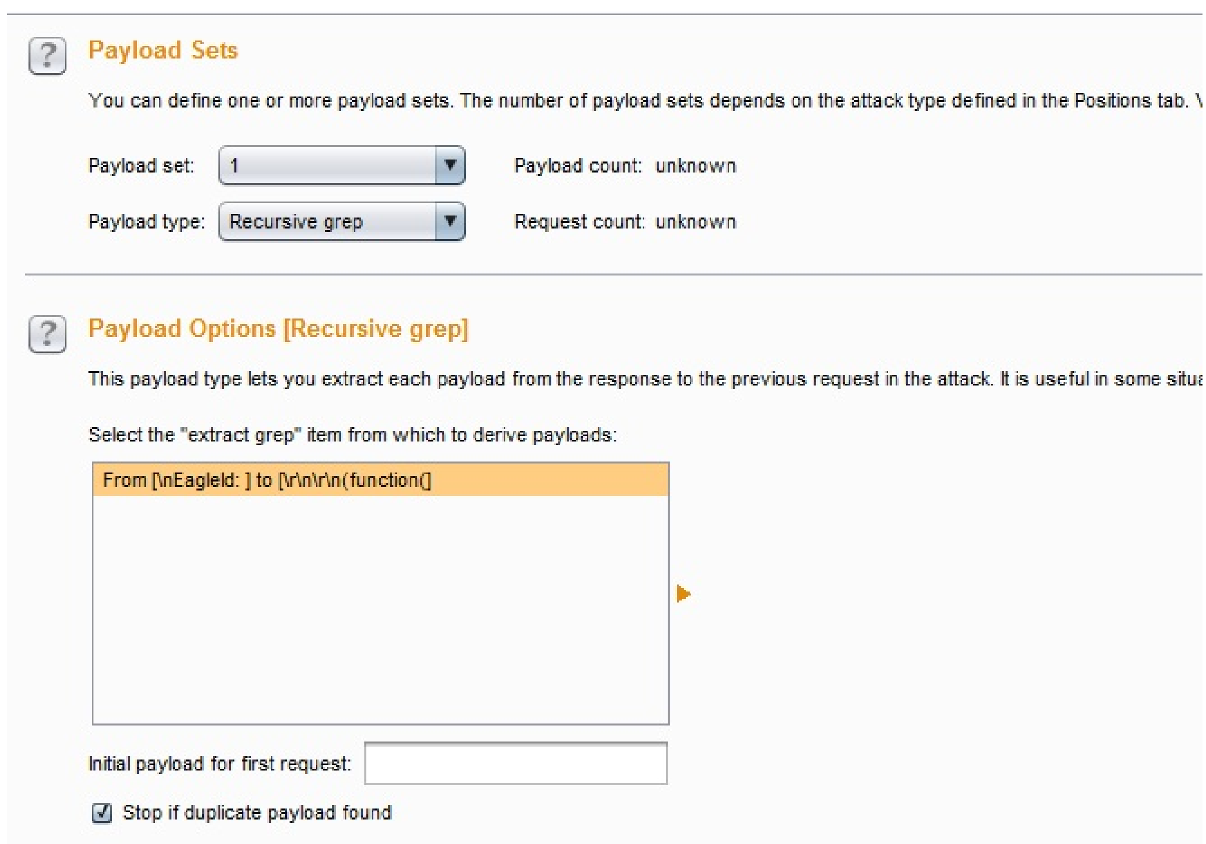
Finished

生成规则由上而下依次是：No change（不改变，使用原始字符串）、To lower case（转为小写字母）、To upper case（转为大写字母）、To Propername（首字母大写，其他小写）、To ProperName（首字母大写，其他不改变），在实际使用中，可以根据自己的使用规则进行勾选设置。

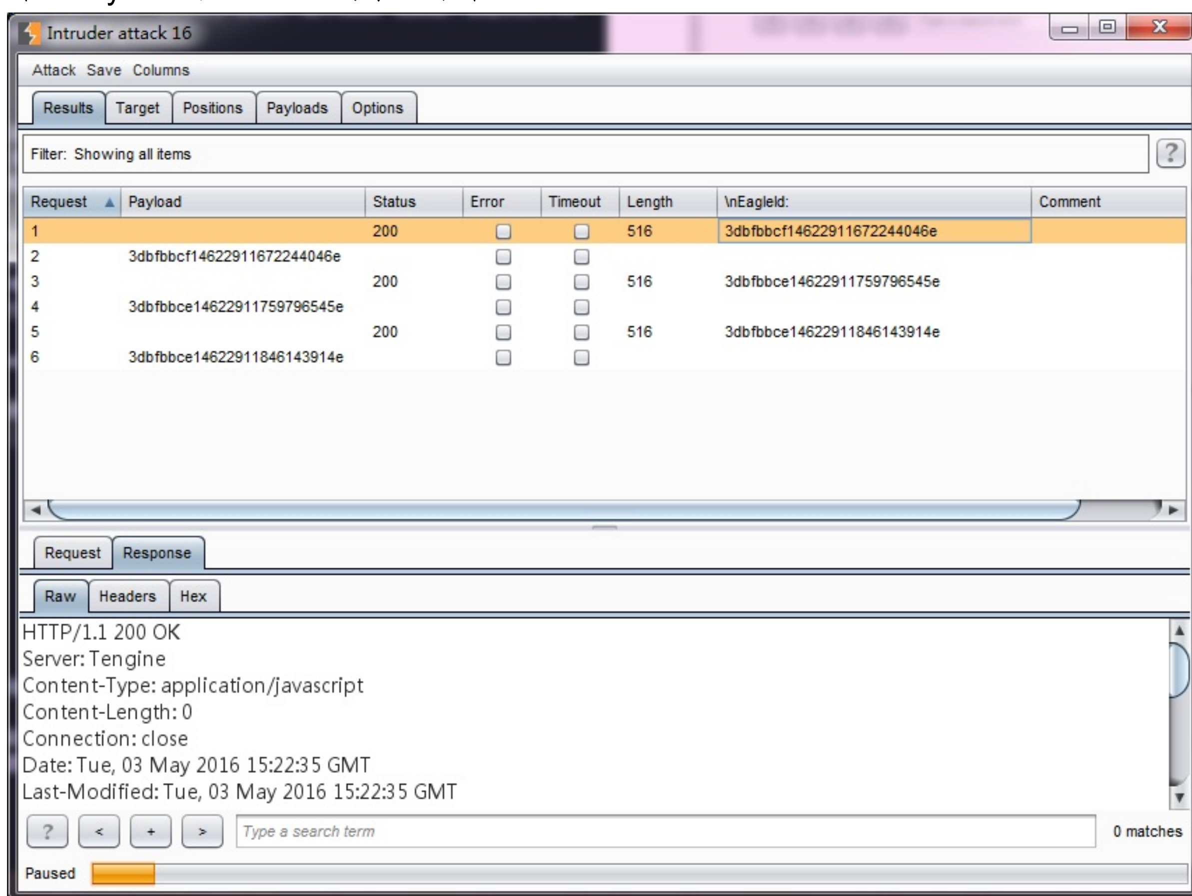
- 递归grep （Recursive grep）——此Payload类型主要使用于从服务器端提取有效数据的场景，需要先从服务器的响应中提取数据作为Payload，然后替换Payload的位置，进行攻击。它的数据来源了原始的响应消息，基于原始响应，在Payload的可选项设置（Options）中配置Grep规则，然后根据grep去提取数据才能发生攻击。比如，我在grep extract 中设置取服务器端的EagleId作为新的Payload值。



点击上图的【OK】按钮之后，完成了Payload的设置。

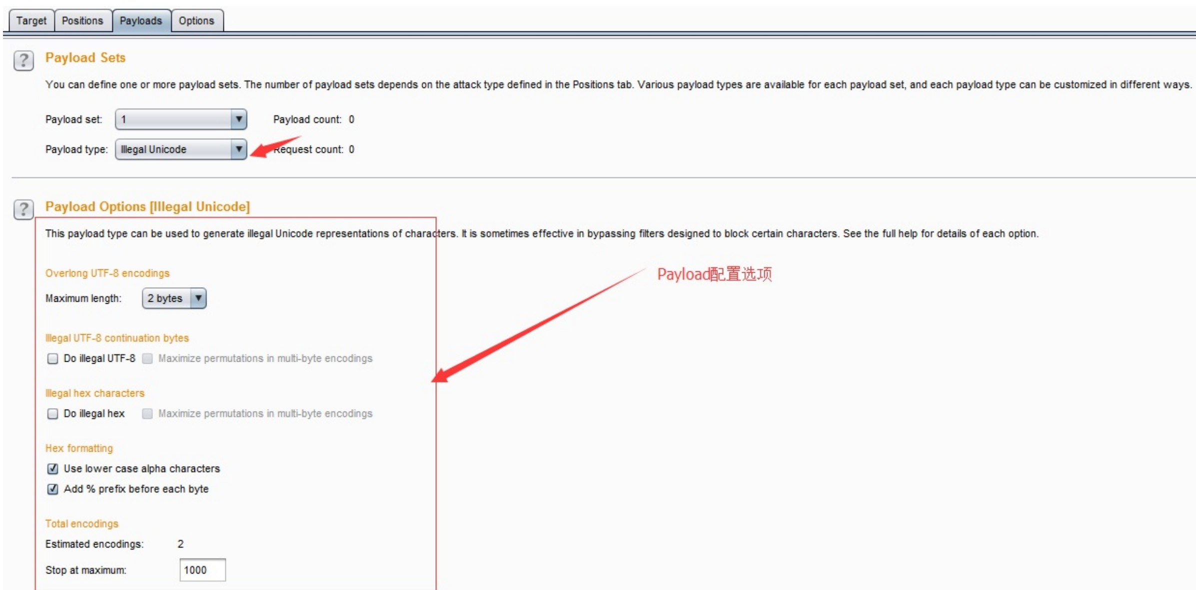


当我发起攻击时，Burp会对每一次响应的消息进行分析，如果提取到了EagleId的值，则作为Payload再发生一次请求。操作图如下：



上图中请求序号为偶数的消息的Payload都是上一次服务器端响应的报文中的EagleId的值。

- 不合法的Unicode编码（Illegal Unicode）—— 在payloads里用指定的不合法Unicode 编码替换字符本身，从这些Payload列表里产生出一个或者多个有效负荷。在尝试回避基于模式匹配的输入验证时，这个有效负荷会有用的，例如，在防御目录遍历攻击时../和..序列的期望编码的匹配。其配置界面如下：



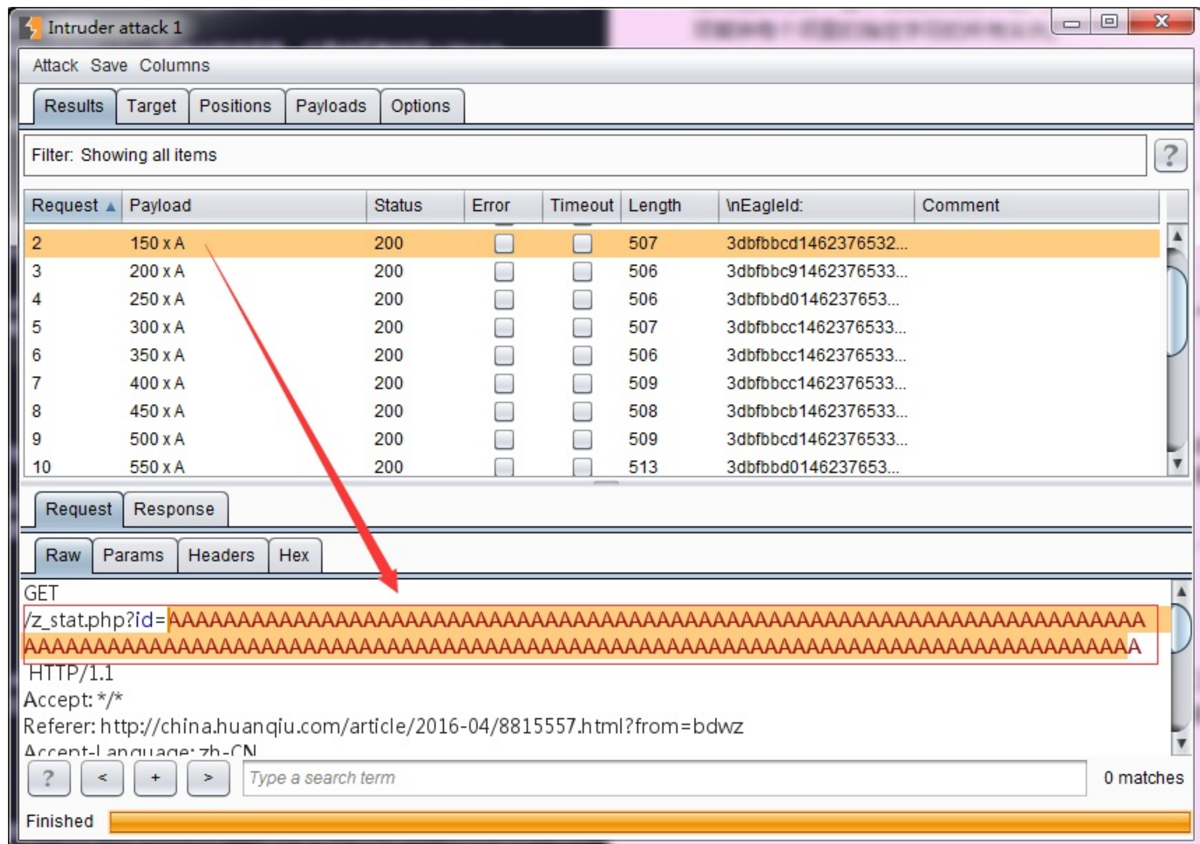
上图中的配置选项主要用来控制不合法编码的生成，各项的含义如下：**maximum overlong UTF-8 length** Unicode 编码允许最多使用 6 字节表示一个字符。使用一种类型就可以正确地表示出(0x00-0x7F) Basic ASCII 字符。然而，使用多字节的Unicode 方案也能表示出它们(如，"overlong"编码)。下拉菜单用来指定是否使用超长编码，以及应该设定的最大使用长度。 **Illegal UTF-8 continuation bytes** 当选择的最大超长 UTF-8 长度为 2 字节以上，这个选项是可用的。 **Do illegal UTF-8** 当使用多字节编码一个字符时，第一个字节后面的字节应该用 10XXXXXX 这样的二进制格式，来指出后续的字节。然而，第一个字节里最有意义的位会指出后面还有多少后续字节。因此，Unicode 编码例程会安全地忽略掉后续字节的前 2 位。这就意味着每个后续字节可能有 3 个非法变种，格式为 00XXXXXX， 01XXXXXX 和 11XXXXXX。如果选中这个选项，则非法 Unicode 有效负荷源会为每个后续字节生成 3 个附加编码。 **Maximize permutations in multi-byte encodings** 如果选择的最大超长 UTF-8 长度为 3 字节以上，并且选中" illegal UTF-8 "这个选项可用 。如果"Maximize permutations in multi-byte encodings"没被选中，则在生产非法变种时，不合法 Unicode 有效负荷源会按顺序处理每个后续字节，为每个后续字节产生 3 个非法变种，并且其他的后续字节不会改变。如果"Maximize permutations in multi-byte encodings"被选中了，不合法 的Unicode 有效负荷源会为后续字节生成所有的非法变种排序 。 如，多个后续字节会同时被修改。在目标系统上回避高级模式匹配控制时，这个功能就会很有用。 **Illegal hex** 这个选择基本上一直可用。当使用超长编码和后续字节的非法变种(如果选中)生成非法编码项列表时，通过修改由此产生的十六进制编码可能会迷惑到某种模式匹配控制。十六进制编码使用字符 A—F 代表十进制 10—15 的值。然而有些十六进制编码会把G解释为 16， H 为 17，等等。因此 0x1G 会被解释为 32。另外，如果非法的十六进制字符使用在一个 2 位数的十六进制编码的第一个位置，则由此产生的编码就会溢出单个字节的大小，并且有些十六进制编码只使用了结果数字的后 8 个有效位，因此 0x1G 会被解码为 257，而那时会被解释为 1。每个合

法的 2 位数的十六进制编码有 4—6 种相关的非法十六进制表示，如果使用的是上面的编码，则这些表示会被解释为同一种十六进制编码。如果”illegal hex”被选中，则非法Unicode 有效负荷源会在非法编码项列表里，生成每个字节的所有可能的非法十六进制编码。 **Maximize permutations in multi-byte encodings** 如果选中的最大超长 UTF-8 长度为 2 字节以上并且“illegal hex”也被选中，则这个选项可用。如果Maximize permutations in multi-byte encodings”没被选中，在生成非法十六进制编码时，非法Unicode 有效负荷源会按顺序处理每个字节。对于每个字节，会生成 4—6 个非法十六进制编码，其他的字节不变。如果Maximize permutations in multi-byte encodings”被选中，则非法 Unicode 有效负荷源会为所有的字节，生成非法十六进制的所有排序。如，多个字节会被同时修改。在目标系统上回避高级模式匹配控制时，这个功能会非常有用。 **add % prefix** 如果选中这个选项，在产生的有效负荷里的每个 2 位数十六进制编码前面，都会插入一个%符号。 **Use lower case alpha characters** 这个选项决定了是否在十六进制编码里使用大小写字母。 **Total encodings** 这个选项为会产生的非法编码数量放置了一个上界，如果大量使用超长编码或者选中了最大列表，这个选项会很有用，因为那会生成大量的非法编码。 **Match / replace in list items** 这个选项用户控制Payload列表中的字符串，它是由匹配字符（Match character）和替换字符编码（Replace with encodings of ）来成对的控制Payload的生成。

当攻击执行时，这个有效负荷源会迭代所有预设项列表，在非法编码集合里，每个预设项替换每个项里的指定字符的所有实例。

- 字符块（Character blocks）——这种类型的Payload是指使用一个给出的输入字符串，根据指定的设置产生指定大小的字符块，表现形式为生成指定长度的字符串。它通常使用了边界测试或缓冲区溢出。

Base string 是指设置原始字符串，Min length是指Payload的最小长度，Max length 是指Payload的最大长度，Step是指生成Payload时的步长。如上图的配置后，生成的Payload如下图所示：



- 数字类型（Number）——这种类型的Payload是指根据配置，生成一系列的数字作为Payload。它的设置界面如下：

Payload set: 1　　Payload count: 0

Payload type: Numbers　　Request count: 0

Payload Options [Numbers]

This payload type generates numeric payloads within a given range and in a specified format.

Number range

Type: Sequential Random

From:

To:

Step:

How many:

Number format

Base: Decimal Hex

Min integer digits:

Max integer digits:

Min fraction digits:

Max fraction digits:

Type表示使用序列还是随机数，**From**表示从什么数字开始，**To**表示到什么数字截止，**Step**表示步长是多少，如果是随机数，则**How many**被激活，表示一共生成多少个

随机数。**Base**表示数字使用十进制还是十六进制形式，**Min integer digits** 表示最小的整数是多少，**Max integer digits**表示最大的整数是多少，如果是10进制，则**Min fraction digits** 表示小数点后最少几位数，**Max fraction digits**表示小数点后最多几位数。

- 日期类型（Dates）——这种类型的Payload是指根据配置，生成一系列的日期。界面如下

Payload set: 1　　Payload count: 1
Payload type: Dates　　Request count: 1

Payload Options [Dates]

This payload type generates date payloads within a given range and in a specified format.

From: 4 May 2016
To: 4 May 2016
Step: 1 Days
Format: 05/2016
E dd.MM.yyyy
Example: 05/2016

其设置选项比较简单，没有什么特别复杂的，不再赘述。至于日期格式，可以选择Burp自己提供的样例格式，也可以自定义，自定义的时候，格式的填写形式如下表所示 | 格式 | 样例| |--------|--------| | E | Sat | | EEEE | Saturday | | d | 7 | | dd | 07 | | M | 6 | | MM | 06 | | MMM | Jun | | MMMM| June | | yy| 16 | | yyyy| 2016 |

- 暴力字典（Brute forcer）——此类Payload生成包含一个指定的字符集的所有排列特定长度的有效载荷，通常用于枚举字典的生成，其设置界面如下：

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

Payload set: 1　　Payload count: 1,679,616
Payload type: Brute forcer　　Request count: 1,679,616

Payload Options [Brute forcer]

This payload type generates payloads of specified lengths that contain all permutations of a specified character set.

Character set: abcdefghijklmnopqrstuvwxyz0123456789
Min length: 4
Max length: 4

Character set 表示生成字典的数据集，从此数据集中抽取字符进行生成。**Min length**表示生成Payload的最小长度，**Max length**表示生成Payload的最大长度。

- 空类型（Null payloads）——这种负载类型产生的Payload，其值是一个空字符串。在攻击时，需要同样的请求反复被执行，在没有任何修改原始请求的场景下此Payload是非常有用的。它可用于各种攻击，例如cookie的序列分析、应用层Dos、或保活会话令牌是在其它的间歇试验中使用。

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each pay customized in different ways.

Payload set: 1 Payload count: 0

Payload type: Null payloads Request count: 0

Payload Options [Null payloads]

This payload type generates payloads whose value is an empty string. With no payload markers configured, this can be used to repeatedly issue the base request unmodified.

Generate payloads

Continue indefinitely

在配置Payload生成方式时，它有两个选项，我们可以指定生成（Generate）多少Payload，也可以设置为一直持续攻击（Continue indefinitely）

- 字符frobber（Character frobber）——这种类型的Payload的生成规律是：依次修改指定字符串在每个字符位置的值，每次都是在原字符上递增一个该字符的ASCII码。它通常使用于测试系统使用了复杂的会话令牌的部件来跟踪会话状态，当修改会话令牌中的单个字符的值之后，您的会话还是进行了处理，那么很可能是这个令牌实际上没有被用来追踪您的会话。其配置界面如图：

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are availat customized in different ways.

Payload set: 1 Payload count: 6

Payload type: Character frobber Request count: 6

Payload Options [Character frobber]

This payload type operates on a string input and modifies the value of each character position in turn. It is useful to quickly test which parts of a long string have a

Operate on: Base value of payload position

Specific string: abcdef

执行后生成的Payload如下图所示：

Results | Target | Positions | Payloads | Options

Filter: Showing all items

Request ▲	Payload	Status	Error	Timeout	Length	Comment
1	bbcdef	200	☐	☐	518	
2	accdef	200	☐	☐	517	
3	abddef	200	☐	☐	524	
4	abceef	200	☐	☐	519	
5	abcdff	200	☐	☐	519	
6	abcdeg	200	☐	☐	516	

- Bit翻转（Bit flipper）——这种类型的Payload的生成规律是：对预设的Payload原始值，按照比特位，依次进行修改。它的设置界面如下图：

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are avail customized in different ways.

Payload set: 1 Payload count: 8

Payload type: Bit flipper Request count: 8

Payload Options [Bit flipper]

This payload type operates on an input and modifies the value of each bit position in turn. It can sometimes be used to meaningfully modify the decrypted value

Operate on: Base value of payload position / Specific string: ab

Format of original data: Literal value / Encoded as ASCII hex

Select bits to flip: 1 (LSB) 3 5 7 / 2 4 6 8 (MSB)

其设置选项主要有：**Operate on** 指定是对Payload位置的原始数据进行Bit翻转还是指定值进行Bit翻转，**Format of original data** 是指是否对原始数据的文本意义进行操作，还是应该把它当作ASCII十六进制，**Select bits to flip**是指选择翻转的Bit位置。 您可以配置基于文本意义进行操作，或基于ASCII十六进制字符串进行翻转。例如，如果原始值是“ab”，基于文本意义的翻转结果是：

```
`b
cb
eb
ib
qb
Ab
!b
¡b
ac
a`
af
aj
ar
aB
a"
a¢
```

如果是基于ASCII十六进制字符串进行翻转，则结果是：

```
aa
a9
af
a3
bb
8b
eb
2b
```

这种类型的Payload类似于字符frobber，但在你需要更细粒度的控制时是有用的。例如，会话令牌或其他参数值使用CBC模式的块密码加密，有可能系统地由前一密码块内修改Bit位以改变解密后的数据。在这种情况下，你可以使用的Bit 翻转的Payload来确定加密值内部修改了个别bit位后是否对应用程序产生影响，并了解应用程序是否容易受到攻击。关于加密模式可以点击阅读这篇文章做进一步的了解。

- 用户名生成器（Username generator）这种类型的Payload主要用于用户名和email帐号的自动生成，其设置界面如下图：

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are customized in different ways.

Payload set: 1　　Payload count: 100 (approx)

Payload type: Username generator　　Request count: 100 (approx)

Payload Options [Username generator]

This payload type lets you configure a list of names or email addresses, and derives potential usernames from these using various common schemes.

Maximum payloads per item: 115

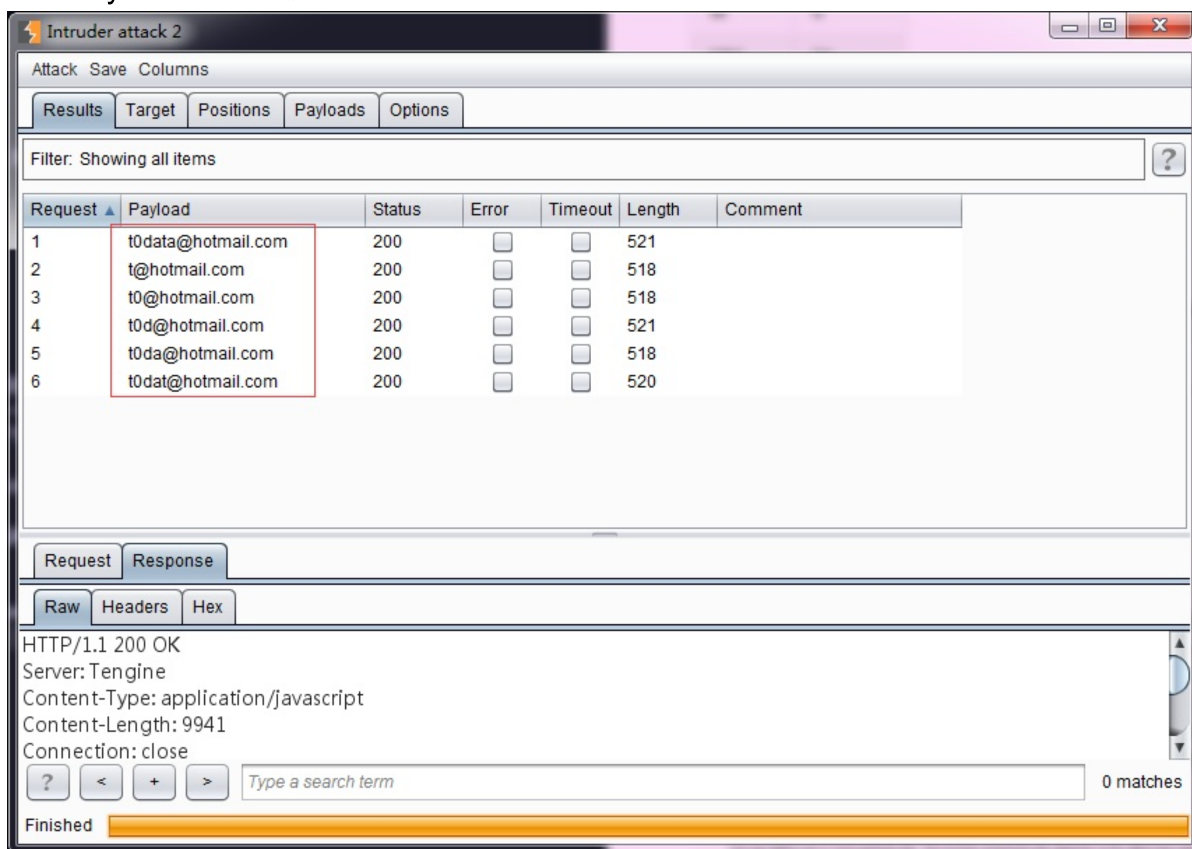
Items (1)

Paste　Load ...　Remove　Clear　Add

t0data@hotmail.com

Enter a new item

如上图所示，我设置了原始值为t0data@hotmail.com,然后执行此Payload生成器，其生成的Payload值如图所示



- ECB 加密块洗牌（ECB block shuffler）——这种类型的Payload是基于ECB加密模式的Payload生成器，关于加密模式可以点击阅读这篇文章做进一步的了解。其原理是因为ECB加密模式中每组64位的数据之间相互独立，通过改变分组数据的位置方式来验证应

用程序是否易受到攻击。其设置界面如下图，Payload的配置参数同上一个Payload类型雷同，就不再赘述。如图：

- Burp Payload生成插件（Extension-generated）——这种类型的Payload是基于Burp插件来生成Payload值，因此使用前必须安装配置Burp插件，在插件里注册一个Intruder payload生成器，供此处调用。其基本设置和使用步骤如下图所示，因后续章节将重点叙述Burp插件，此处不再展开。

- Payload复制（Copy other payload）——这种类型的Payload是将其他位置的参数复制到Payload位置上，作为新的Payload值，通常适用于多个参数的请求消息中，它的使用场景可能是： 1.两个不同的参数需要使用相同的值，比如说，用户注册时，密码设置会输入两遍，其值也完全一样，可以使用此Payload类型。 2.在一次请求中，一个参数的值是基于另一个参数的值在前端通过脚本来生成的值，可以使用此Payload类型。 它的设置界

面和参数比较简单，如下图所示，其中Payload位置的索引值就是指向图中Payload set的值。

Payload位置和攻击类型

首先我们来看看Payload位置（Payload positions）选项卡的设置界面：

从上图中我们可以看出，Payload位置的设置是基于Http请求的原始消息作为母板，使用一对§字符来标记出Payload的位置，在这两个号直接包含了母板文本内容。 当我们已经把一个Payload在请求消息的特殊位置上时标明后，发起攻击时，Burp Intruder 就把一个Payload值放置到给出的特殊位置上，替换 §符号标示的整个位置。如上图中的参数id后面的 §符号之间的标明的是Payload位置1，name后面的 §符号之间标明的是Payload位置2，这个值对应于Payload设置中的Payload set的值。 我们可以在消息编辑器中间对Payload位置进行编辑，它主要是由右侧的四个按钮来控制的。

- 【Add §】——在当前光标的位置添加一个Payload位置标志
- 【Clear §】——清除所有Payload位置标志或者清除选中的Payload位置标志

- 【Auto §】——对消息内容中可能需要标志的参数做一个猜测，标志为Payload位置，自动设置完之后再做人工的选择，确定哪些位置是需要传入Payload的。目前Burp支持自动选择的参数类型有： 1.URL请求参数 2.Body参数 3.cookie参数 4.复合型参数属性，比如文件上传时候的文件名 5.XML数据 6.JSON数据 虽然Burp默认是支持自动标志这些类型的参数作为Payload位置，但如果是针对于像XML或JSON的节点属性值的，还是需要手工指定。
- 【Refresh】——刷新消息内容中带有颜色的部分。
- 【Clear】——清除消息编辑器中所有内容。

在消息编辑器的上方，有一个下拉选择框，攻击类型（Attack Type）。Burp Intruder支持使用Payload进行多种方式的模拟攻击，目前只要有以下4种。

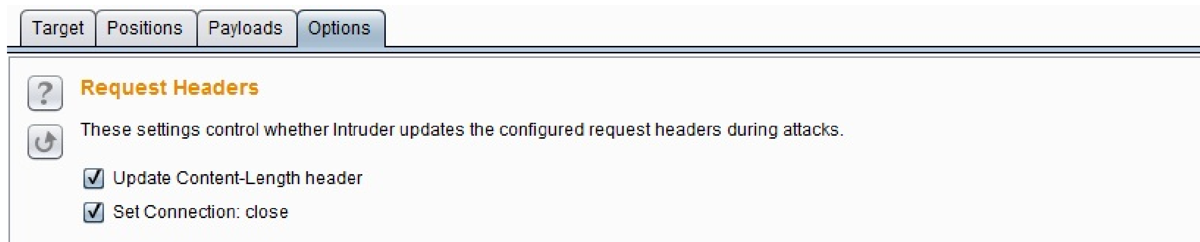
- 狙击手模式（Sniper）——它使用一组Payload集合，依次替换Payload位置上（一次攻击只能使用一个Payload位置）被§标志的文本（而没有被§标志的文本将不受影响），对服务器端进行请求，通常用于测试请求参数是否存在漏洞。
- 攻城锤模式（Battering ram）——它使用单一的Payload集合，依次替换Payload位置上被§标志的文本（而没有被§标志的文本将不受影响），对服务器端进行请求，与狙击手模式的区别在于，如果有多个参数且都为Payload位置标志时，使用的Payload值是相同的，而狙击手模式只能使用一个Payload位置标志。
- 草叉模式（Pitchfork ）——它可以使用多组Payload集合，在每一个不同的Payload标志位置上（最多20个），遍历所有的Payload。举例来说，如果有两个Payload标志位置，第一个Payload值为A和B，第二个Payload值为C和D，则发起攻击时，将共发起两次攻击，第一次使用的Payload分别为A和C，第二次使用的Payload分别为B和D。
- 集束炸弹模式（Cluster bomb） 它可以使用多组Payload集合，在每一个不同的Payload标志位置上（最多20个），依次遍历所有的Payload。它与草叉模式的主要区别在于，执行的Payload数据Payload组的乘积。举例来说，如果有两个Payload标志位置，第一个Payload值为A和B，第二个Payload值为C和D，则发起攻击时，将共发起四次攻击，第一次使用的Payload分别为A和C，第二次使用的Payload分别为A和D，第三次使用的Payload分别为B和C，第四次使用的Payload分别为B和D。

可选项设置（**Options**）

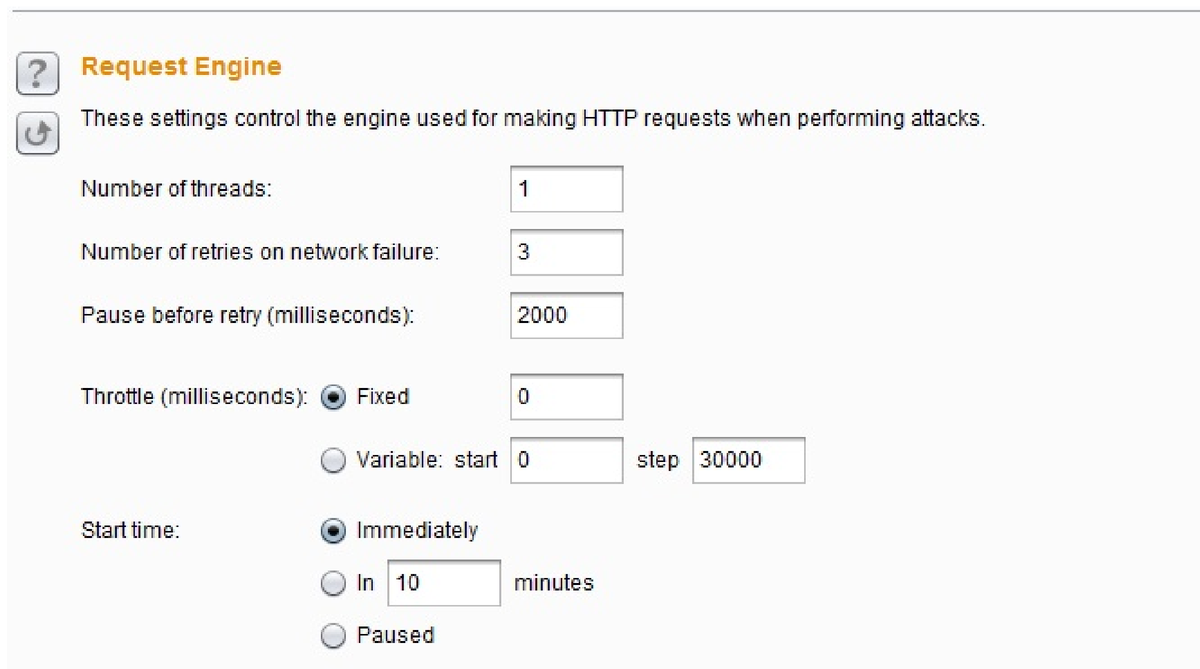
可选项设置主要包括请求消息头设置、请求引擎设置、攻击结果设置、grep match, grep extract, grep payloads,以及重定向设置。在使用中，你可以在攻击前进行设置，也可以在攻击过程中做这些选项的调整。

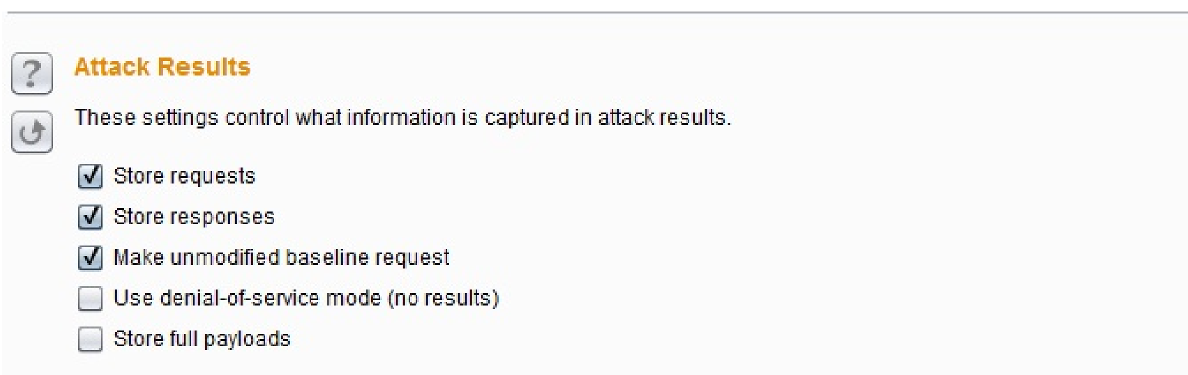
- 请求消息头设置（Request Headers）——这个设置主要用来控制请求消息的头部信息，它由**Update Content-Length header**和**Set Connection: close**两个选项组成。其中

Update Content-Length header如果被选中，Burp Intruder在每个请求添加或更新Content-Length头为该次请求的HTTP体的长度正确的值。这个功能通常是为插入可变长度的Payload到模板的HTTP请求的主体的攻击中，如果没有指定正确的值，则目标服务器可能会返回一个错误，可能会到一个不完整的请求做出响应，或者可能会无限期地等待请求继续接收数据。**Set Connection: close**如果被选中，表示Burp Intruder在每个请求消息中添加或更新值为“关闭”的连接头，这将更迅速地执行。在某些情况下（当服务器本身并不返回一个有效的Content-Length或Transfer-Encoding头），选中此选项可能允许攻击。



- 请求引擎设置（Request Engine）——这个设置主要用来控制Burp Intruder攻击，合理地使用这些参数能更加有效地完成攻击过程。它有如下参数：**Number of threads**并发的线程数，**Number of retries on network failure** 网络失败时候重试次数，**Pause before retry**重试前的暂停时间间隔（毫秒），**Throttle between requests** 请求延时（毫秒），**Start time**开始时间，启动攻击之后多久才开始执行。





- Grep Match——这个设置主要用来从响应消息中提取结果项，如果匹配，则在攻击结果中添加的新列中标明，便于排序和数据提取。比如说，在密码猜测攻击，扫描诸如“密码不正确”或“登录成功”，可以找到成功的登录;在测试SQL注入漏洞，扫描包含“ODBC”，“错误”等消息可以识别脆弱的参数。



其选项有**Match type**表示匹配表达式还是简单的字符串，**Case sensitive match**是否大小写敏感，**Exclude HTTP headers**匹配的时候，是否包含http消息头。

- Grep Extract——这些设置可用于提取响应消息中的有用信息。对于列表中配置的每个项目，Burp会增加包含提取该项目的文本的新结果列。然后，您可以排序此列（通过单击列标题）命令所提取的数据。此选项是从应用数据挖掘有用的，能够支持广泛的攻击。例如，如果你是通过一系列文档ID的循环，可以提取每个文档寻找有趣的项目的页面标题。如果您发现返回的其他应用程序用户详细信息的功能，可以通过用户ID重复和检索有关用户寻找管理帐户，甚至密码。如果“遗忘密码”的功能需要一个用户名作为参数，并

有关用户寻找管理帐户，甚至密码。如果“遗忘密码”的功能需要一个用户名作为参数，并返回一个用户配置的密码提示，您可以通过共同的用户名列表运行和收获的所有相关密码的提示，然后直观地浏览列表寻找容易被猜到密码。

Grep - Extract

These settings can be used to extract useful information from responses into the attack results table.

Extract the following items from responses:

Add

Edit

Remove

Duplicate

Up

Down

Clear

Maximum capture length: 100

- Grep Payloads——这些设置可用于提取响应消息中是否包含Payload的值，比如说，你想验证反射性的XSS脚本是否成功，可以通过此设置此项。当此项设置后，会在响应的结果列表中，根据Payload组的数目，添加新的列，显示匹配的结果，你可以通过点击列标题对结果集进行排序和查找。

Grep - Payloads

These settings can be used to flag result items containing reflections of the submitted payload.

Search responses for payload strings

Case sensitive match

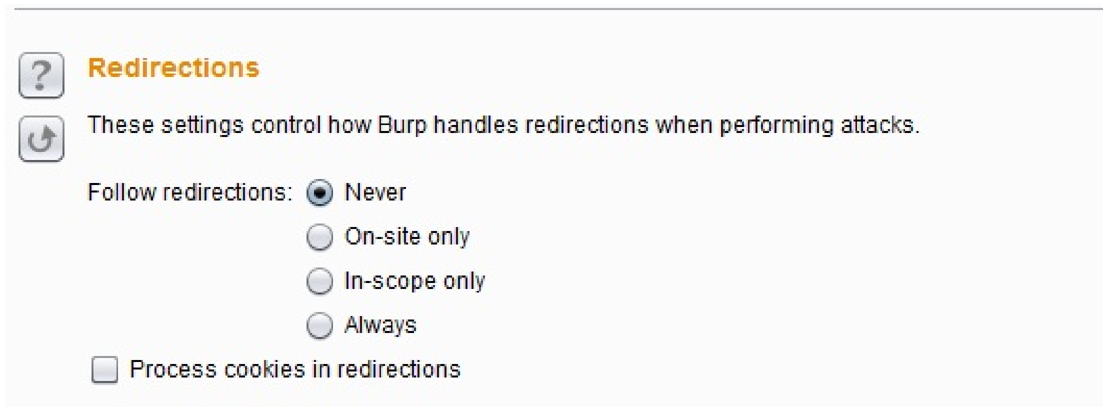
Exclude HTTP headers

Match against pre-URL-encoded payloads

其设置项跟上一个类似，需要注意的是**Match against pre-URL-encoded payloads**，如果你在请求消息时配置了 URL-encode payloads ，则这里表示匹配未编码之前的Payload值，而不是转码后的值。

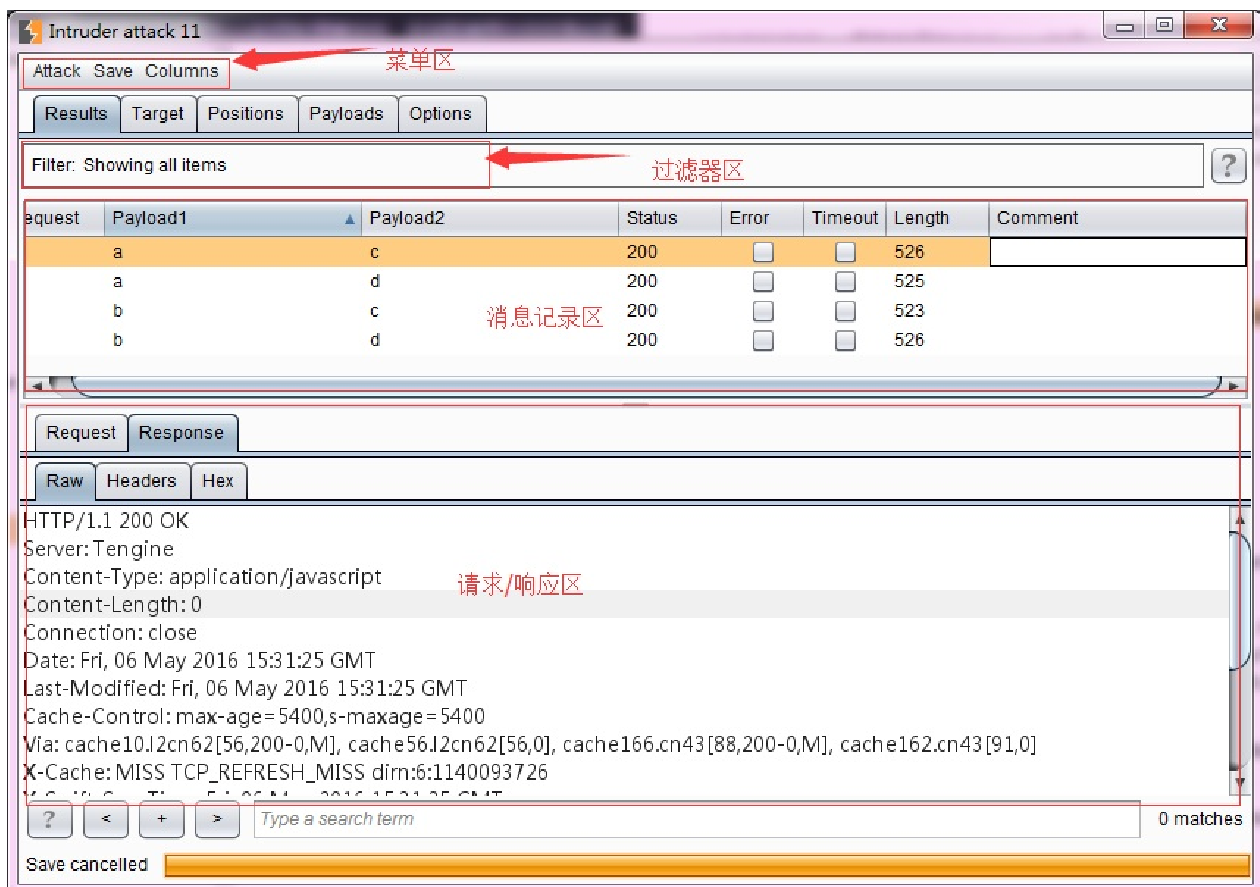
- 重定向（Redirections）——这些设置主要是用来控制执行攻击时Burp如何处理重定向，在实际使用中往往是必须遵循重定向，才能实现你的攻击目的。例如，在密码猜测攻击，每次尝试的结果可能是密码错误会重定向响应到一个错误消息提示页面，如果密码正确会重定向到用户中心的首页。 但设置了重定向也可能会遇到其他的问题，比如说，在某些情况下，应用程序存储您的会话中初始请求的结果，并提供重定向响应时检索此值，这时可能有必要在重定向时只使用一个单线程攻击。也可能会遇到，当你设置重定

向，应用程序响应会重定向到注销页面，这时候，按照重定向可能会导致您的会话被终止时。 因其设置选项跟其他模块的重定向设置基本一致，此处就不再重叙。



Intruder 攻击和结果分析

一次攻击的发起，通常有两种方式。一种是你在Burp Intruder里设置了Target, Positions, Payloads and Options ，然后点击【Start attack】启动攻击；另一种是你打开一个之前保存的预攻击文件，然后点击【Start attack】启动攻击。无论是哪种方式的攻击发起，Burp都将弹出攻击结果界面。在攻击的过程中，你也可以修改攻击配置，或者做其他的操作。攻击结果的界面如下图所示：



从上图我们可以看出，其界面主要又菜单区、过滤器、Payload执行结果消息记录区、请求/响应消息区四大部分组成。

- 菜单区 包含Attack菜单、Save菜单、Columns菜单。 **Attack**菜单主要用来控制攻击行为的，你可以暂停攻击（**pause**）、恢复攻击（**resume**）、再次攻击（**repeat**）。 **Save**菜单主要用来保存攻击的各种数据，**Attack** 保存当前攻击的副本，下次可以从此文件进行再次攻击；**Results table**保存攻击的结果列表，相当于图中的Payload执行结果消息记录区数据，当然你可以选择行和列进行保存，只导出你需要的数据；**Server responses** 保存所有的服务器响应消息；**Attack configuration**保存当前的攻击配置信息。 **Columns**菜单主要用来控制消息记录区的显示列。如果某个列被选中，则在Payload执行结果消息记录区显示，反之则不显示。

- 过滤器 ——可以通过查询条件、服务器响应的状态码、注释过Payload执行结果消息记录区的信息进行过滤。

- Payload执行结果消息记录区，又称结果列表（Results Table），记录Payload执行时请求和响应的所有信息，它包含的列有： 请求序列——显示请求的序列号，如果配置了记录未修改的请求消息母板，则会在第一个进行记录。 **Payload**位置——狙击手模式下会记录 **Payload**值——如果有多个Payload，则存在多个列 **HTTP** 状态码——服务器响应状态码 请求时间——执行攻击的时间 响应时间——开始接受到响应时间，单位为毫秒。 响应完成时间——响应完成的时间，单位为毫秒。 网络错误——Payload执行时是否发生网络问题 超时情况——等待应答响应过程中，是否发生网络超时 长度——响应消息的长度 **Cookie**——任何的Cookie信息 **Grep**——如果设置了Grep 匹配、Grep 提取、Grep Payload，则会有多个列显示匹配的信息 重定向——如果配置了重定向，则显示 注释——消息记录的注释信息

- 请求/响应消息区——参考Proxy章节的相关叙述。

在对攻击结果的分析中，你可以通过单击任一列标题（单击标题循环通过升序排序，降序排序和未排序）重新排序表的内容。有效地解释攻击的结果的一个关键部分是定位有效的或成功的服务器响应，并确定生成这些请求。有效的应答通常可以通过以下中的至少一个存在差异： 1.不同的HTTP状态代码 2.不同长度的应答 3.存在或不存在某些表达式 4.错误或超时的发生 5.用来接收或完成响应时间的差异 比如说，在URL路径扫描过程中，对不存在的资源的请求可能会返回“404未找到”的响应，或正确的URL会反馈的“200 OK”响应。或者在密码猜测攻击，失败的登录尝试可能会产生一个包含“登录失败”关键字“200 OK”响应，而一个成功的登录可能会生成一个“302对象移动”的反应，或不同的“200 OK”响应页面。

每一个渗透测试人员，对Burp Intruder 攻击结果的分析方式可能会存在差异，这主要源于个人水平的不同和经验的不同。在实战中，只有慢慢尝试，积累，才能通过快速地对攻击结果的分析获取自己关注的重要信息。

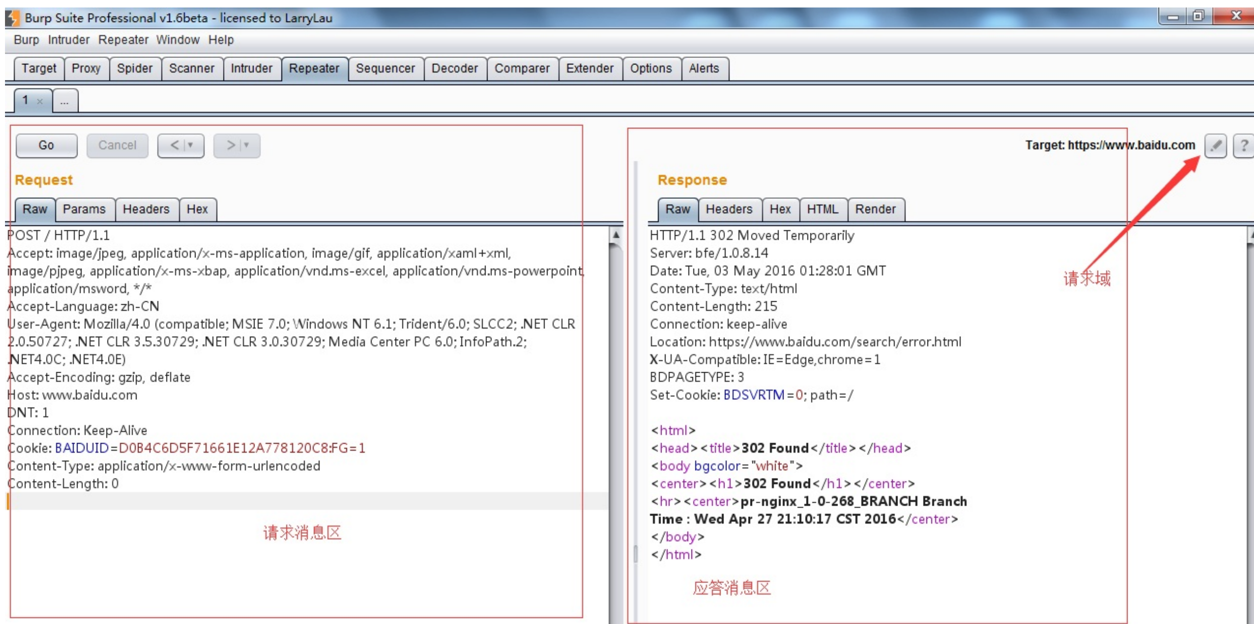
第九章 如何使用Burp Repeater

Burp Repeater作为Burp Suite中一款手工验证HTTP消息的测试工具，通常用于多次重放请求响应和手工修改请求消息的修改后对服务器端响应的消息分析。本章我们主要学习的内容有：

- Repeater的使用
- 可选项设置（Options）

Repeater的使用

在渗透测试过程中，我们经常使用Repeater来进行请求与响应的消息验证分析，比如修改请求参数，验证输入的漏洞；修改请求参数，验证逻辑越权；从拦截历史记录中，捕获特征性的请求消息进行请求重放。Burp Repeater的操作界面如下图所示：



请求消息区为客户端发送的请求消息的详细信息，Burp Repeater为每一个请求都做了请求编号，当我们在请求编码的数字上双击之后，可以修改请求的名字，这是为了方便多个请求消息时，做备注或区分用的。在编号的下方，有一个【GO】按钮，当我们对请求的消息编辑完

之后，点击此按钮即发送请求给服务器端。服务器的请求域可以在target处进行修改，如上图所示。

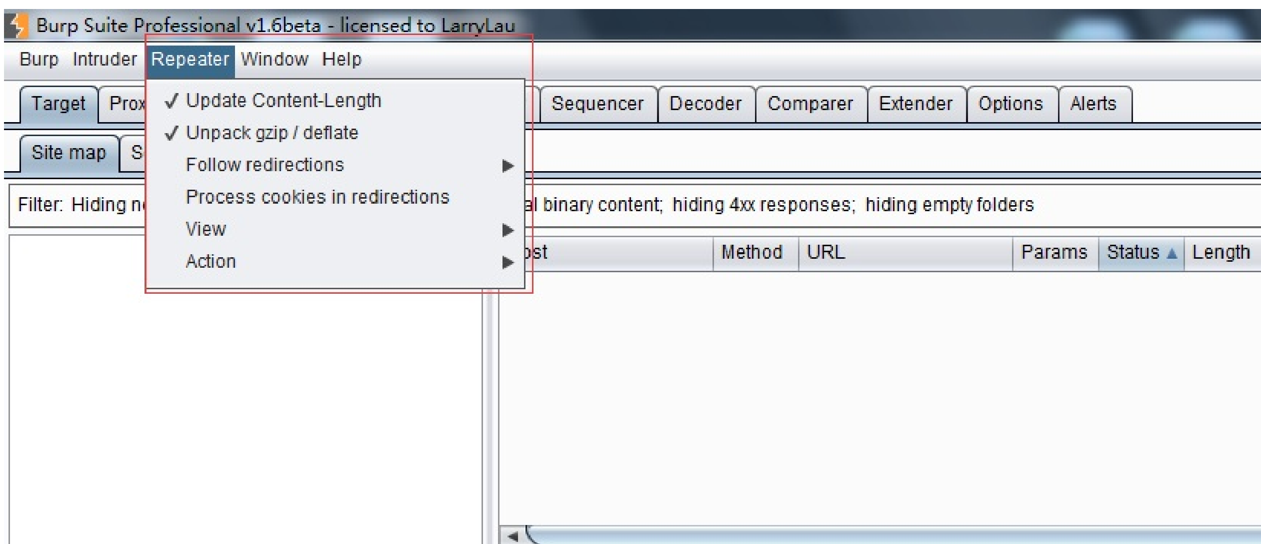


应答消息区为对应的请求消息点击【GO】按钮后，服务器端的反馈消息。通过修改请求消息的参数来比对分析每次应答消息之间的差异，能更好的帮助我们分析系统可能存在的漏洞。

在我们使用Burp Repeater时，通常会结合Burp的其他工具一起使用，比如Proxy的历史记录，Scanner的扫描记录、Target的站点地图等，通过其他工具上的右击菜单，执行【Send to Repeater】，跳转到Repeater选项卡中，然后才是对请求消息的修改以及请求重放、数据分析与漏洞验证。

可选项设置（**Options**）

与Burp其他工具的设置不同，Repeater的可选项设置菜单位于整个界面顶部的菜单栏中，如图所示：



其设置主要包括以下内容：

- 更新Content-Length
  这个选项是用于控制Burp是否自动更新请求消息头中的Content-Length
- 解压和压缩（Unpack gzip / deflate ） 这个选项主要用于控制Burp是否自动解压或压缩服务器端响应的内容

- 跳转控制（Follow redirections） 这个选项主要用于控制Burp是否自动跟随服务器端作请求跳转，比如服务端返回状态码为302，是否跟着应答跳转到302指向的url地址。 它有4个选项，分别是永不跳转（Never），站内跳转（On-site only ）、目标域内跳转（In-scope only）、始终跳转（Always），其中永不跳转、始终跳转比较好理解，站内跳转是指当前的同一站点内跳转；目标域跳转是指target scope中配置的域可以跳转；
- 跳转中处理Cookie（Process cookies in redirections ） 这个选项如果选中，则在跳转过程中设置的Cookie信息，将会被带到跳转指向的URL页面，可以进行提交。
- 视图控制（View） 这个选项是用来控制Repeater的视图布局
- 其他操作（Action） 通过子菜单方式，指向Burp的其他工具组件中。

第十章 如何使用Burp Sequencer

Burp Sequencer作为Burp Suite中一款用于检测数据样本随机性质量的工具，通常用于检测访问令牌是否可预测、密码重置令牌是否可预测等场景，通过Sequencer的数据样本分析，能很好地降低这些关键数据被伪造的风险。本章我们主要学习的内容有：

- Sequencer使用步骤
- 可选项设置（Options）

Sequencer使用步骤

Burp Sequencer作为一款随机数分析的工具，在分析过程中，可能会对系统造成不可预测的影响，在你不是非常熟悉系统的情况下，建议不要在生产环境进行数据分析。它的使用步骤大体如下：1.首先，确认Burp Suite安装正确，并配置好浏览器代理，正常运行。2.从Burp Proxy的历史日志记录中，寻找token或类似的参数，返回右击弹出上下文菜单，点击【Send to Sequencer】。

Target | Proxy | Spider | Scanner | Intruder | Repeater | Sequencer | Decoder | Comparer | Extender | Options | Alerts | Bradamsa

Intercept | HTTP history | WebSockets history | Options

Filter: Hiding CSS, image and general binary content

# ▲	Host	Method	URL	Params	Edited	Status	Length	MIME type	Extension
829	http://home.baidu.com	GET	/resource/r/home/flash.js	☐	☐	200	1218	script	js
832	http://home.baidu.com	GET	/resource/r/home/banner.swf	☐	☐	200	13300	flash	swf
842	http://home.baidu.com	GET	/inddata.xml	☐	☐	200	1278	XML	xml
848	http://hm.baidu.com	GET	/h.js?b9a77820b2fa17d7223b50a...	☑	☐	200	23459	script	js
849	http://home.baidu.com				☐	200	6264	HTML	html
854	http://hm.baidu.com				☐	200	23459	script	js
859	http://se.360.cn				☐	200	2624	text	ini
861	http://www.baidu.com				☐	200	49697	HTML	
863	http://www.baidu.com				☐	200	10475	script	js
864	http://hm.baidu.com				☐	200	23459	script	js

http://home.baidu.com/product/product.html
Add to scope
Spider from here
Do an active scan
Do a passive scan
Send to Intruder Ctrl+I
Send to Repeater Ctrl+R
Send to Sequencer
Send to Comparer (request)
Send to Comparer (response)
Show response in browser
Request in browser ▶
Engagement tools ▶
Show new history window
Add comment
Highlight ▶
Delete item

Request | Response

Raw | Headers | Hex | HTML

HTTP/1.1 200 OK
Date: Sat, 07 May 2016 12:01:
Server: Apache/2.2.31 (Unix)
Last-Modified: Mon, 28 Mar 2
ETag: "ea05b3-1779-52f1695
Accept-Ranges: bytes
Content-Length: 6009
Connection: close
Content-Type: text/html

3.进入Burp Sequencer的Live Capture面板，选中刚才发送过来的记录，点击【Configure】配置需要分析的token或者参数。

Target | Proxy | Spider | Scanner | Intruder | Repeater | Sequencer | Decoder | Comparer | Extender | Options | Alerts | Bradamsa

Live capture | Manual load | Analysis options

Select Live Capture Request

Send requests here from other tools to configure a live capture. Select the request to use, configure the other options below, then click "Start live capture".

Remove

Clear

#	Host	Request
6	http://home.baidu.com	GET /product/product.html HTTP/1.1Acce...

Start live capture

Token Location Within Response

Select the location in the response where the token appears.

Cookie:

Form field:

Custom location: From [\nETag: "] to ["\r\nAccept-] Configure

4.在弹出的参数配置对话框中，选中参数的值，点击【OK】按钮，完成参数设置。

Define custom token location

Define the location of the token within the response. Selecting the item in the response panel will create a suitable configuration automatically. You can also modify the configuration manually to ensure it works effectively.

Define start and end

Start after expression: \nETag: "

Start at offset: 137

End at delimiter: "\r\nAccept-

End at fixed length: 25

Extract from regex group

\nETag\: "(.*?)"\r\nAccept\-

Case sensitive

Exclude HTTP headers Update config based on selection below Refetch response

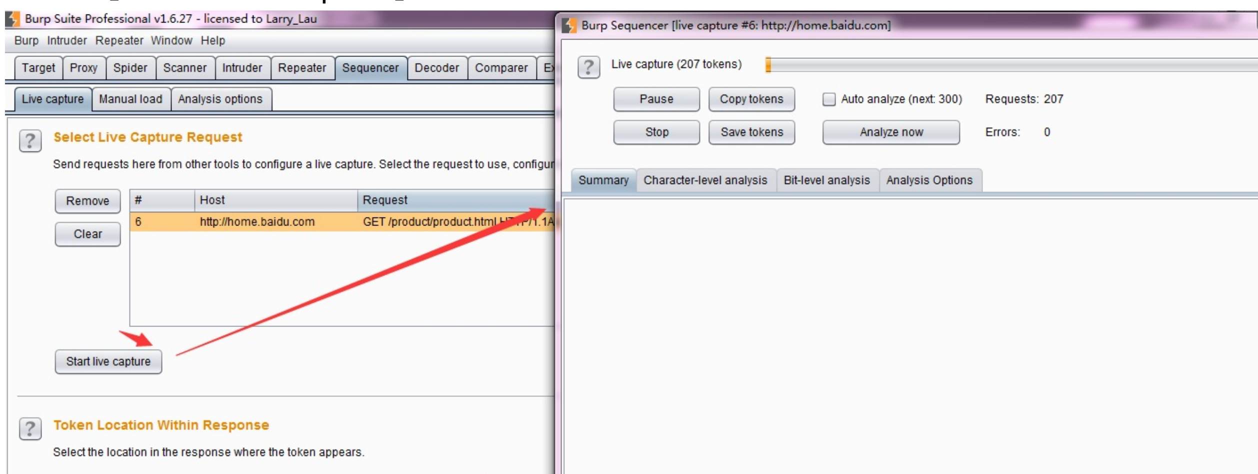
```
HTTP/1.1 200 OK
Date: Sat, 07 May 2016 12:01:26 GMT
Server: Apache/2.2.31 (Unix)
Last-Modified: Mon, 28 Mar 2016 07:08:54 GMT
ETag: "ea05b3-1779-52f1695e2ad80"
Accept-Ranges: bytes
Content-Length: 6009
Connection: close
Content-Type: text/html

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
```

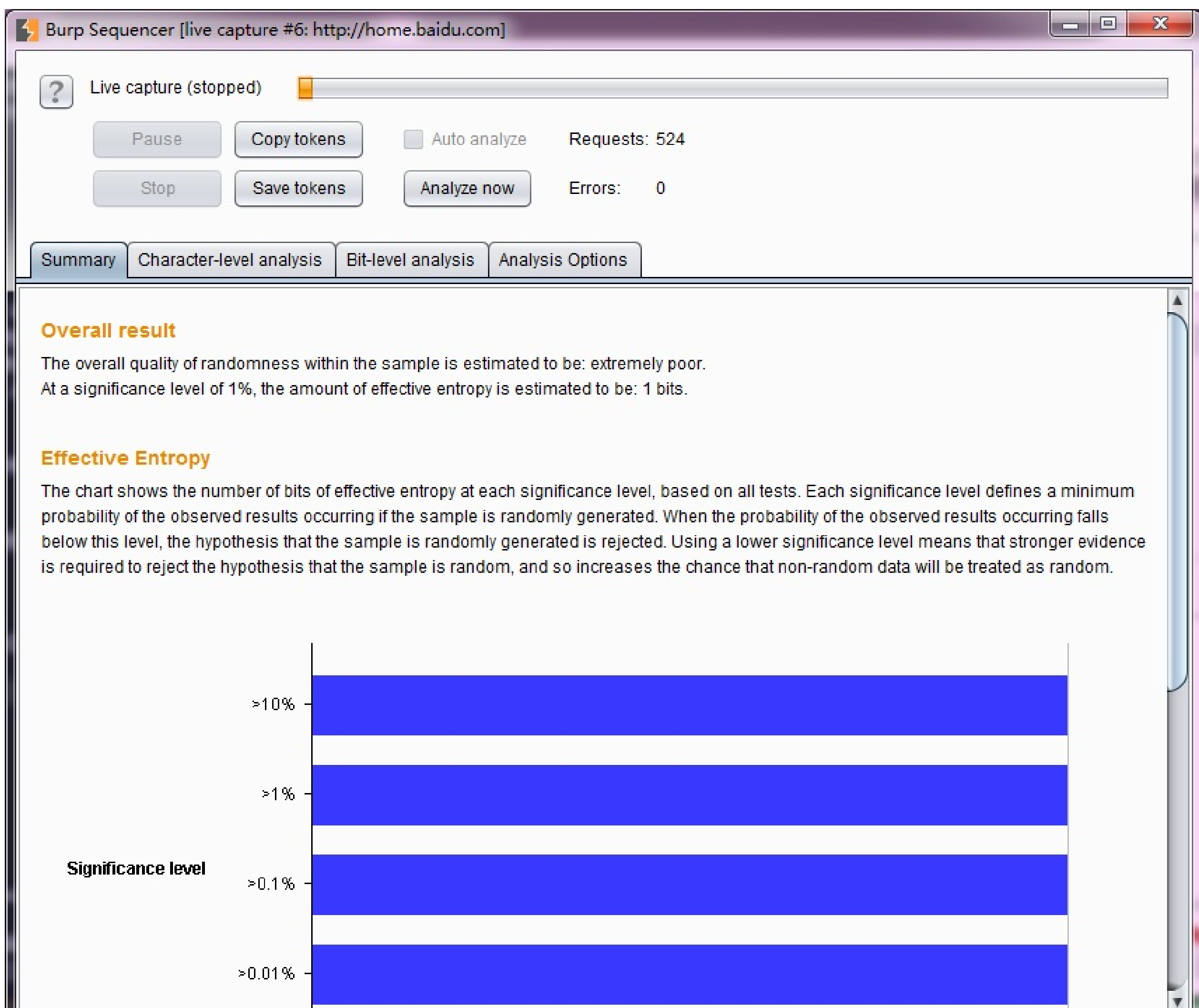
? < + > Type a search term 0 matches

OK Cancel

5.点击【Select Live Capture】，开始进行参数值的获取。



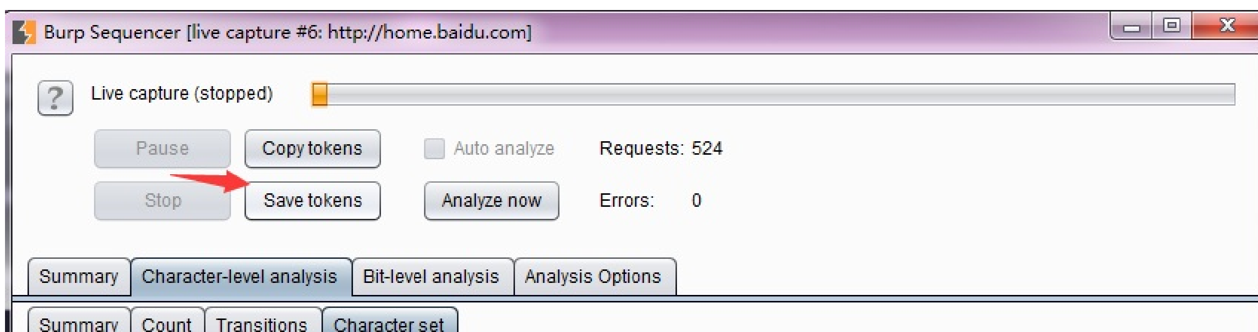
6.当抓取的参数值总数大于100时，点击【pause】或者【stop】，这时可以进行数据分析，点击【Analyze now】即进行数据的随机性分析。



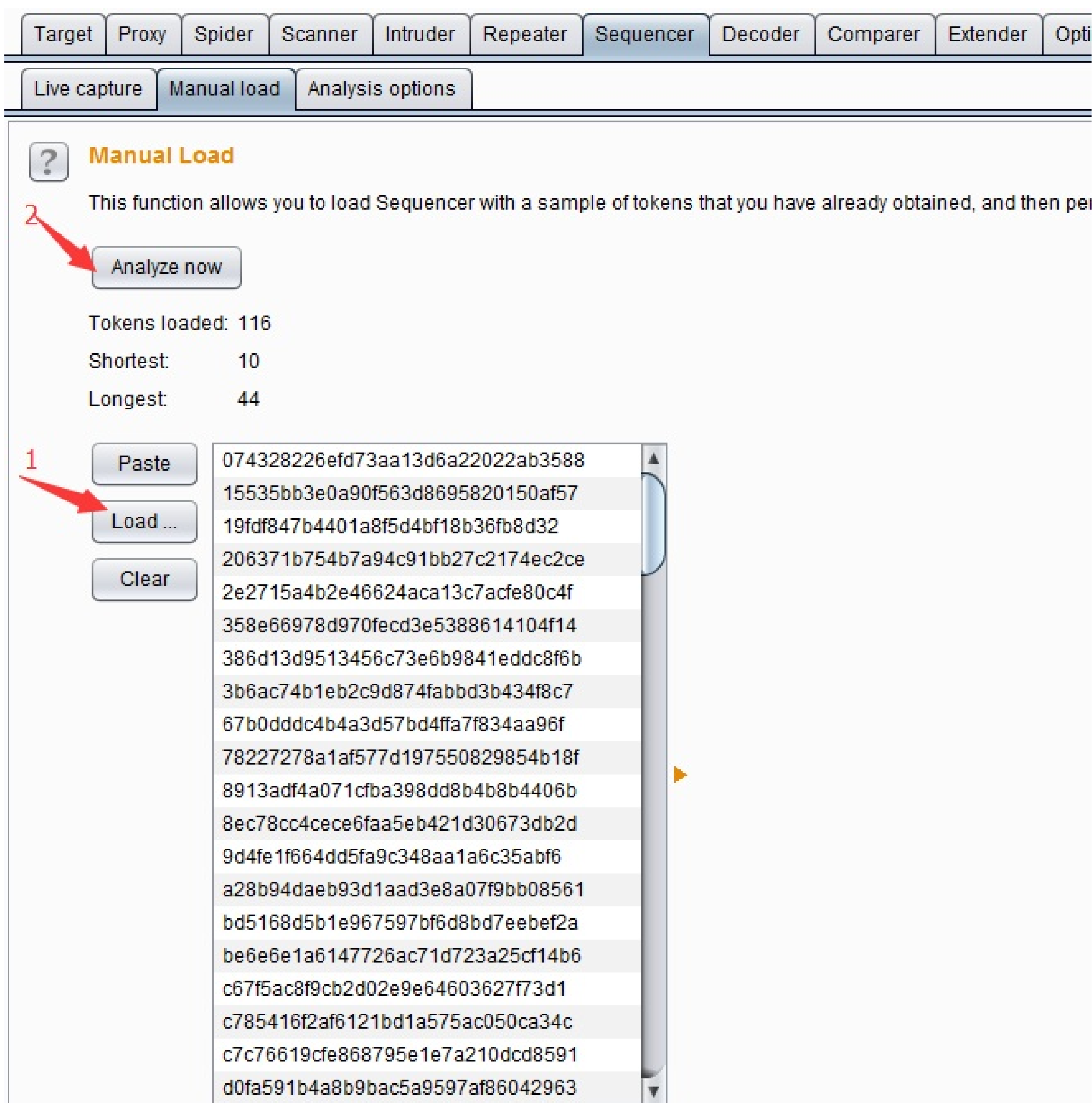
7.等分析结束，则可以看到分析结果的各种图表。

Burp Sequencer [live capture #6: http://home.baidu.com]
Live capture (stopped)
Pause
Copy tokens
Auto analyze
Requests: 524
Stop
Save tokens
Analyze now
Errors: 0
Summary
Character-level analysis
Bit-level analysis
Analysis Options
Summary
Count
Transitions
Character set
Character Set
The chart shows the size of the character set used at each position. This number is the count of different characters that appear at each position within the sample data.
2
1
0

8.当然，我们也可以把获取的数据保存起来，下一次使用的时候，从文件加载参数，进行数据分析。如下图保存数据。

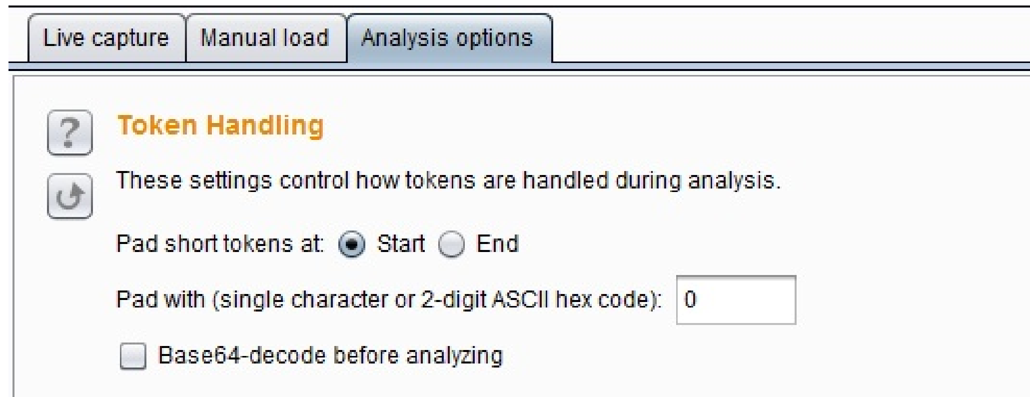


9.当我再次使用时，直接加载数据进行分析即可。



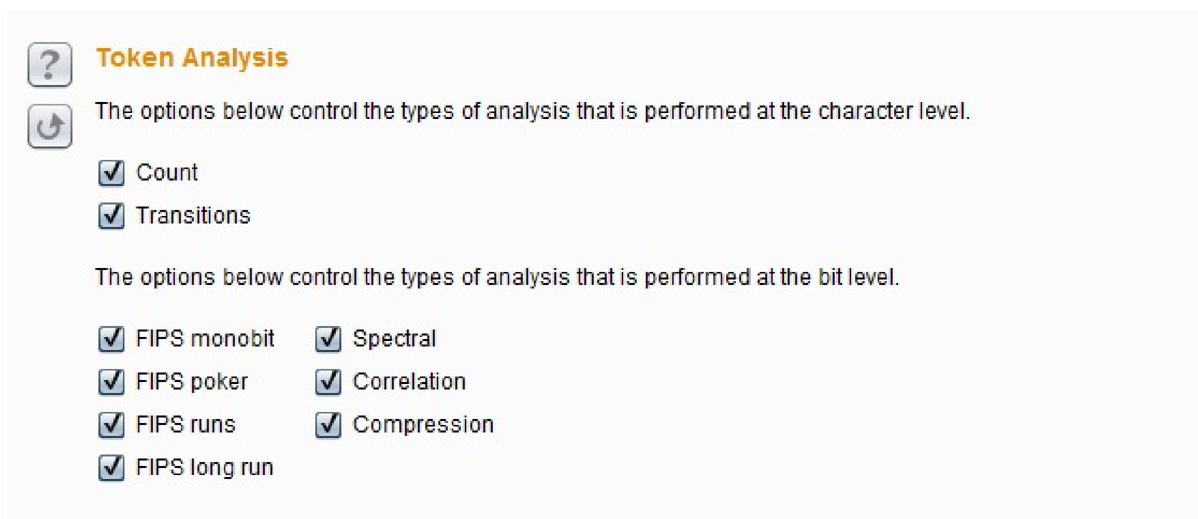
可选项设置（**Analysis Options**）

分析可选项设置的目的主要是为了控制token或者参数，在进行数据分析过程中，需要做什么样的处理，以及做什么类型的随机性分析。它主要由令牌处理（**Token Handling**）和令牌分析（**Token Analysis**）两部分构成。

- 令牌处理**Token Handling**主要控制令牌在数据分析中如何被处理，它的设置界面如下图所示：

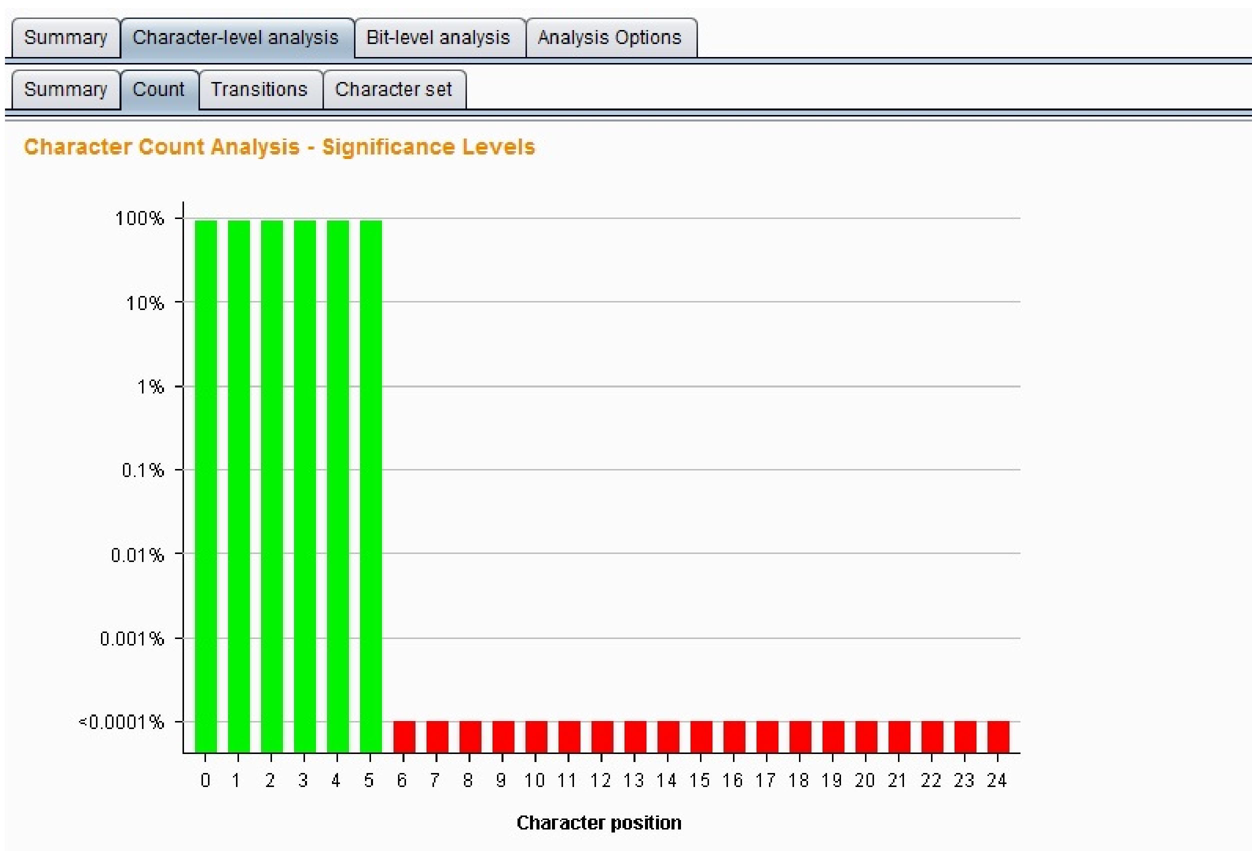
其中**Pad short tokens at start / end** 表示如果应用程序产生的令牌是具有可变长度的，那么这些令牌在数据分析前都需要被填充，以便于进行的统计检验。你可以选择是否填充在开始位置或每个令牌的结束位置。在大多数情况下，在开始位置填充是最合适。 **Pad with** 表示你可以指定将用于填充的字符。在大多数情况下，数字或ASCII十六进制编码的令牌，用“0”填充是最合适的。 **Base64-decode before analyzing**表示在数据分析是否进行base64解码，如果令牌使用了base64编码的话，则需要勾选此项。

- 令牌分析**Token Analysis**主要用来控制对数据进行随机性分析的类型，我们可以选择多个分析类型，也可以单独启用或禁用每个字符类型级和字节级测试。有时候，执行与启用所有分析类型进行初步分析后，再禁用某些分析类型，以便更好地了解令牌的特点，或隔离由样品表现任何不寻常的特性。其设置界面如下：



其中上面两个选项是控制数据分析的字符类型级，它包含**Count**和**Transitions**。 **Count**是指分析在令牌内的每个位置使用的字符的分布，如果是随机生成的样本，所用字符的分布很可能是大致均匀的。在每个位置上分析统计令牌是随机产生的分布的概率。其分析结果图表如下所示：

其中上面两个选项是控制数据分析的字符类型级，它包含**Count**和**Transitions**。 **Count**是指分析在令牌内的每个位置使用的字符的分布，如果是随机生成的样本，所用字符的分布很可能是大致均匀的。在每个位置上分析统计令牌是随机产生的分布的概率。其分析结果图表如下所示：



Transitions是指分析样品数据中的连续符号之间的变化。如果是随机生成的样品，出现在一个给定的位置上的字符是同样可能通过在该位置使用的字符中的任一项中的下一个标志的改变。在每个位置上统计分析令牌随机产生到变化的概率。其分析结果图表如下所示：

下面的几项设置是用于控制数据分析的字节级测试，它比字符级测试功能更强大。启用字节级分析中，每个令牌被转换成一组字节，与设置在每个字符位置的字符的大小决定的比特的总数。它包含的测试类型有以下七种。

FIPS monobit test——它测试分析0和1在每个比特位置的分配，如果是随机生成的样本，1和0的数量很可能是大致相等。Burp Sequencer 记录每个位是通过还是没通过FIPS试验观测。值得注意的是，FIPS测试正式规范假定样本总数为20000个时。如果你希望获得的结果与该FIPS规范一样严格的标准，你应该确保达到20000个令牌的样本。 其分析结果图表如下所示：

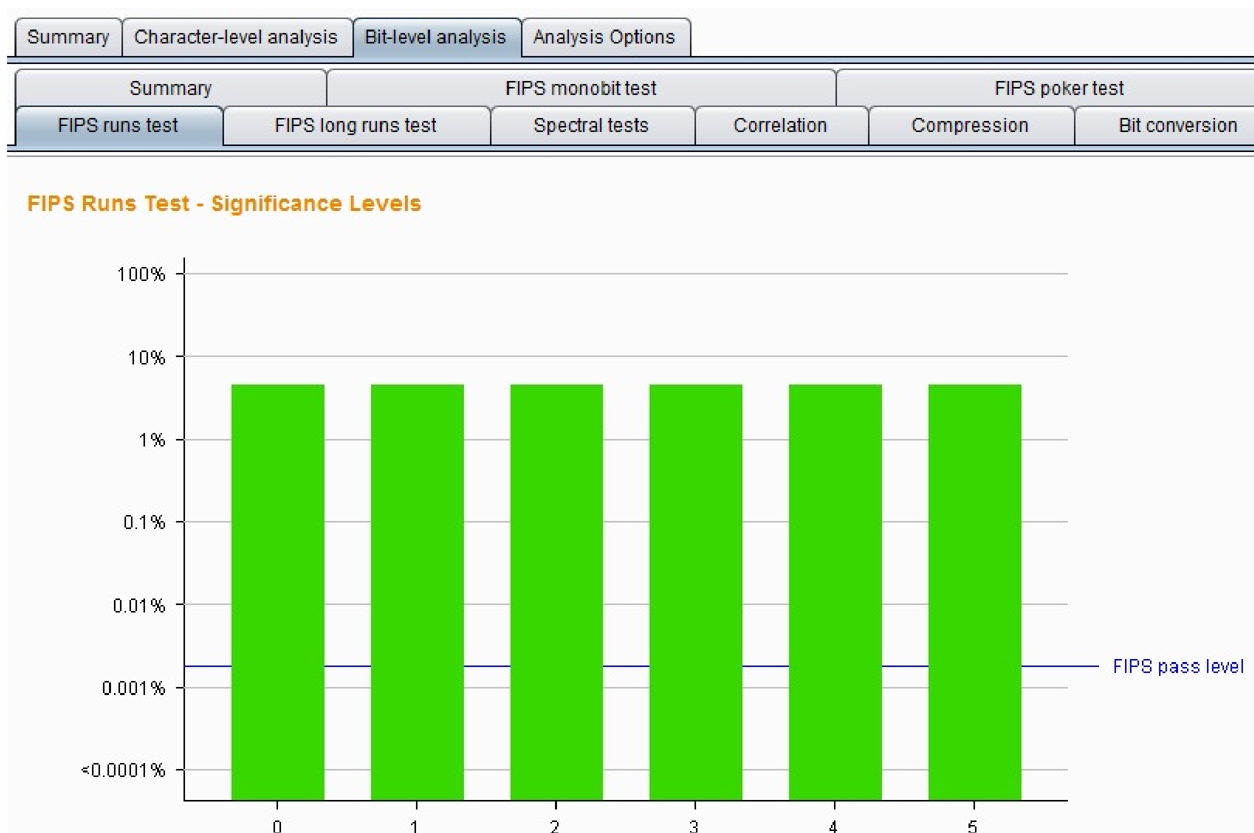


FIPS poker test—— 该测试将j比特序列划分为四个连续的、非重叠的分组，然后导出4个数，计算每个数字出现16个可能数字的次数，并采用卡方校验来评估数字的分布。如果样品是随机生成的，这个数字的分布可能是近似均匀的。在每个位置上，通过该测试方式，分析令牌是随机产生的分布的概率。其分析结果图表如下所示：

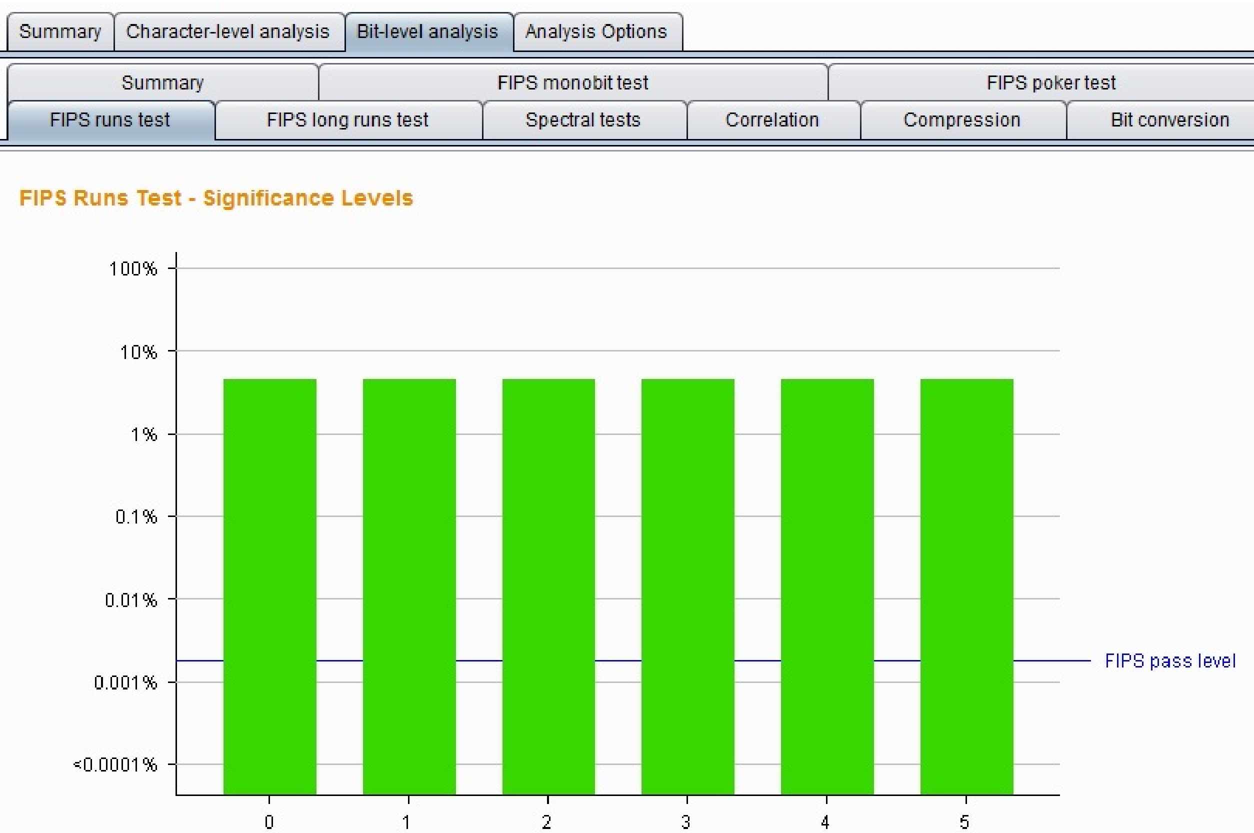
与该FIPS规范一样严格的标准，你应该确保达到20000个令牌的样本。 其分析结果图表如下所示：



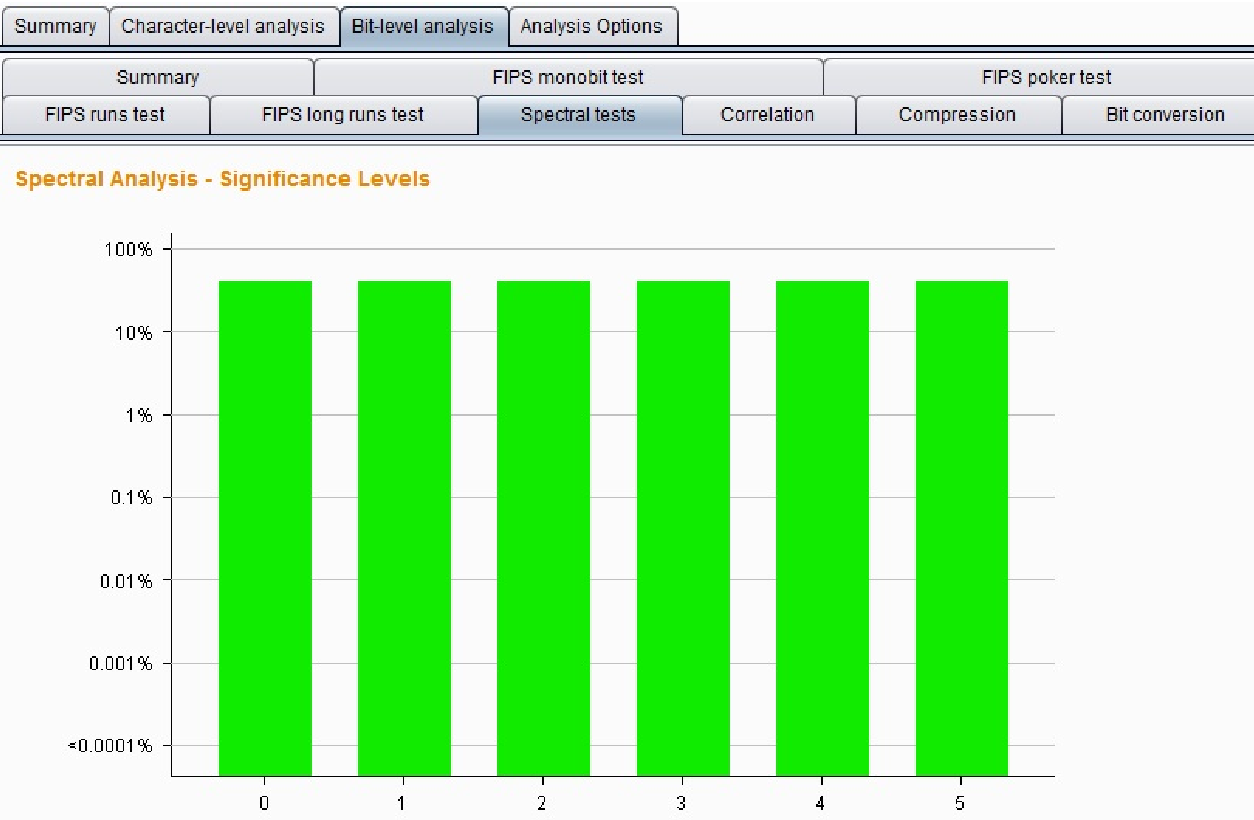
FIPS runs tests —— 该测试将具有相同值的连续的比特序列在每一个位置进行划分成段，然后计算每一个段的长度为1，2，3，4，5，和6以及6以上。如果样品是随机生成的，那么这些段的长度很可能是由样本集的大小所确定的范围之内。在每个位置上，使用该分析方法，观察令牌是随机生成的概率。其分析结果图表如下所示：



FIPS long runs test —— 这个测试将有相同值的连续的比特序列在每一个位置进行划分成段，统计最长的段。如果样品是随机生成的，最长的段的数量很可能是由样本集的大小所确定的范围之内。在每个位置上，使用此分析方法，观察令牌是随机产生的最长段的概率。其分析结果图表如下所示：

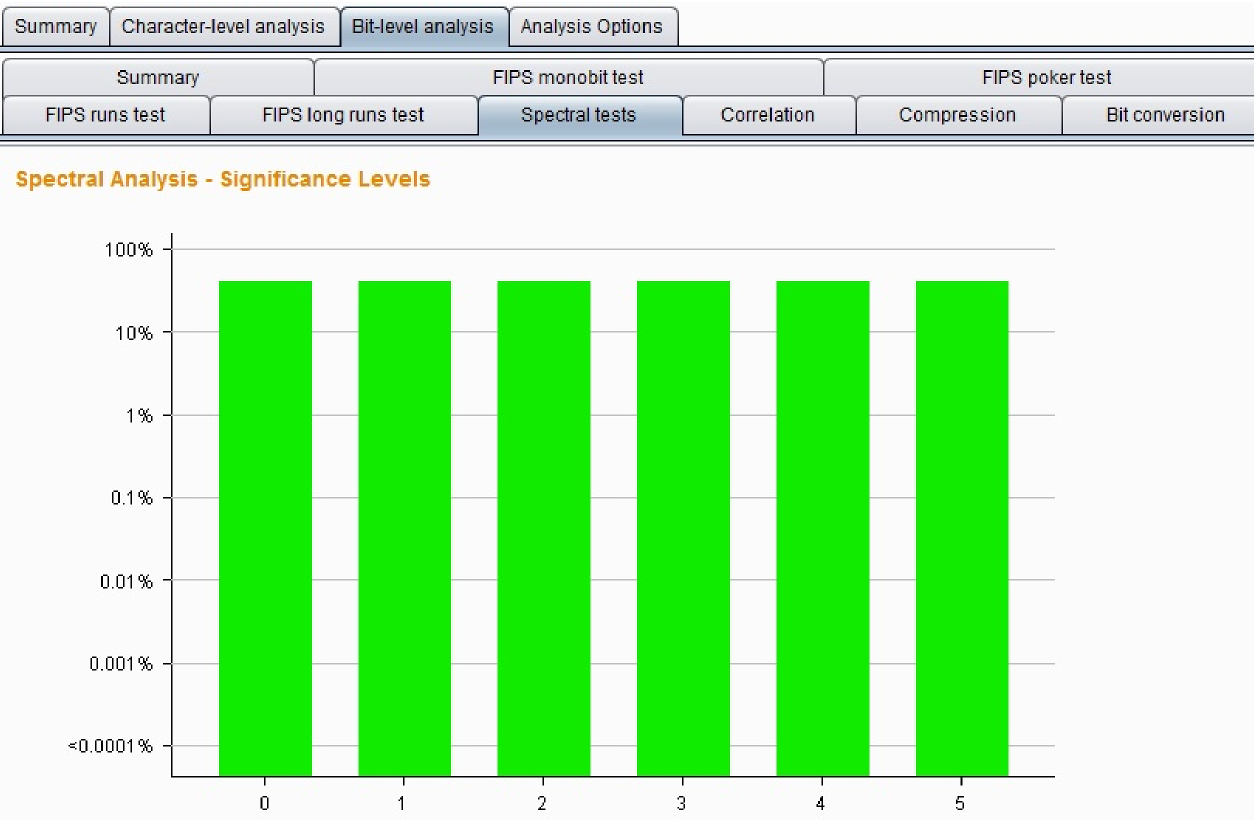

Summary
Character-level analysis
Bit-level analysis
Analysis Options
Summary
FIPS monobit test
FIPS poker test
FIPS runs test
FIPS long runs test
Spectral tests
Correlation
Compression
Bit conversion
FIPS Runs Test - Significance Levels
100%
10%
1%
0.1%
0.01%
0.001%
<0.0001%
0
1
2
3
4
5
FIPS pass level

可能是非随机的。在每个位置，使用此种分析方法，观察令牌是随机发生的概率。其分析结果图表如下所示：

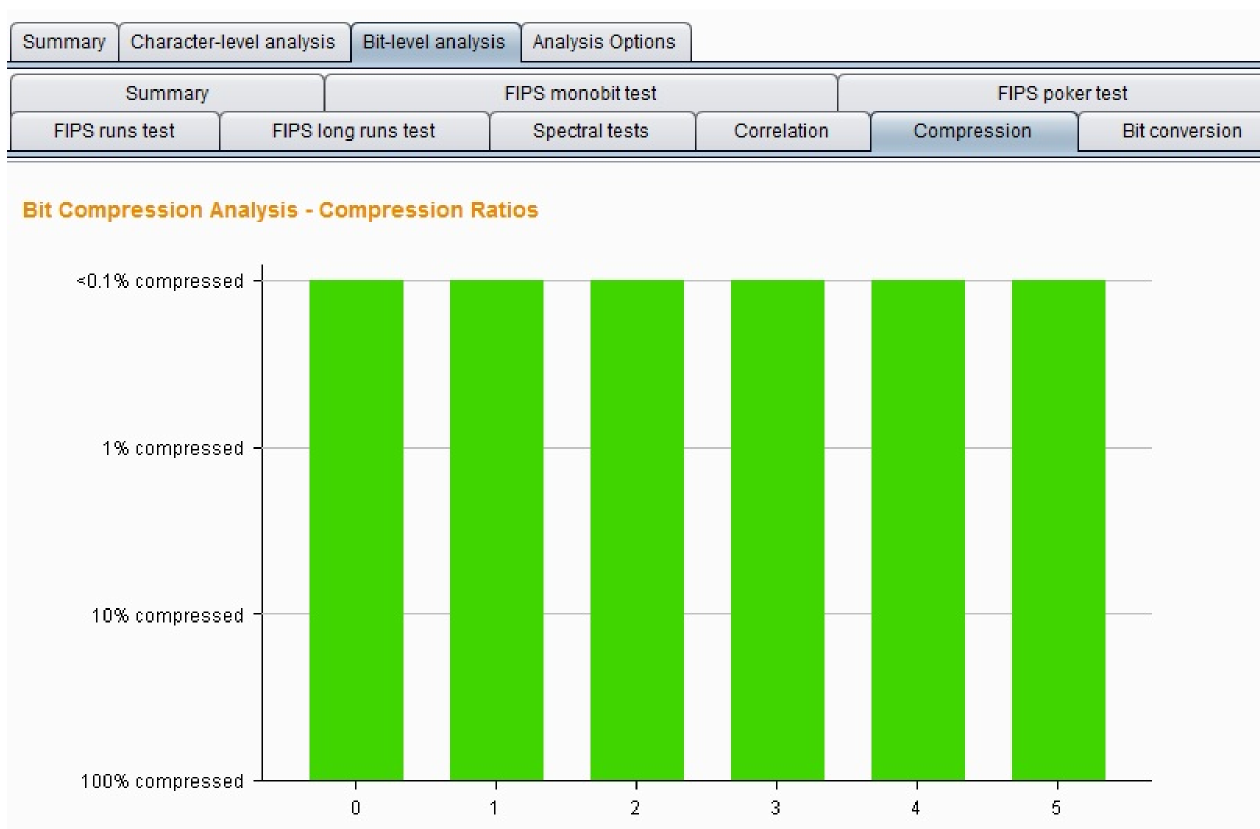


Correlation test ——比较每个位置具有相同值的令牌样本与每一个位置具有不同值的短令牌样本之间的熵，以测试在令牌内部的不同的比特位置中的值之间的任何统计学显著关系。如果样品是随机生成的，在给定的比特位置处的值是同样可能伴随着一个或一个零在任何其它位的位置。在每个位置上，使用此种分析方法，观察令牌是随机生成的可能性。为了防止任

可能是非随机的。在每个位置，使用此种分析方法，观察令牌是随机发生的概率。其分析结果图表如下所示：



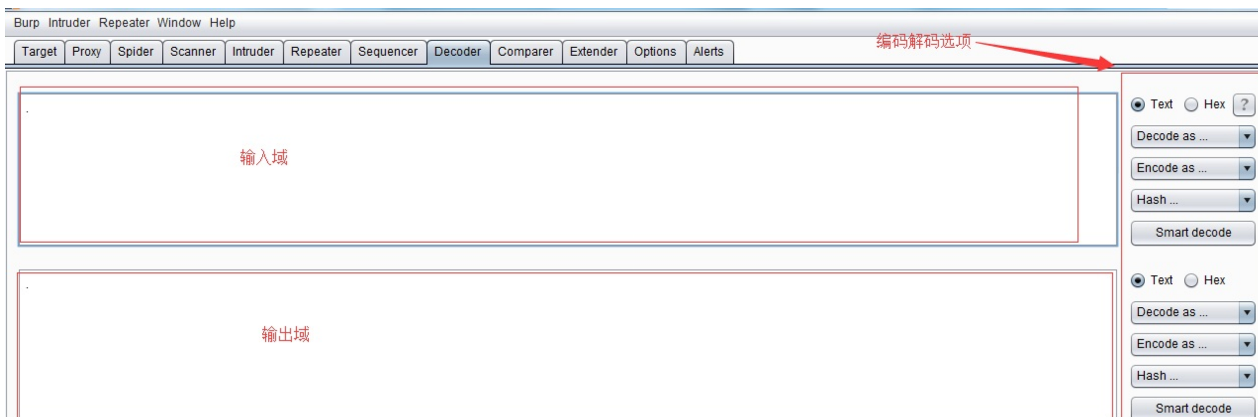
Correlation test ——比较每个位置具有相同值的令牌样本与每一个位置具有不同值的短令牌样本之间的熵，以测试在令牌内部的不同的比特位置中的值之间的任何统计学显著关系。如果样品是随机生成的，在给定的比特位置处的值是同样可能伴随着一个或一个零在任何其它位的位置。在每个位置上，使用此种分析方法，观察令牌是随机生成的可能性。为了防止任



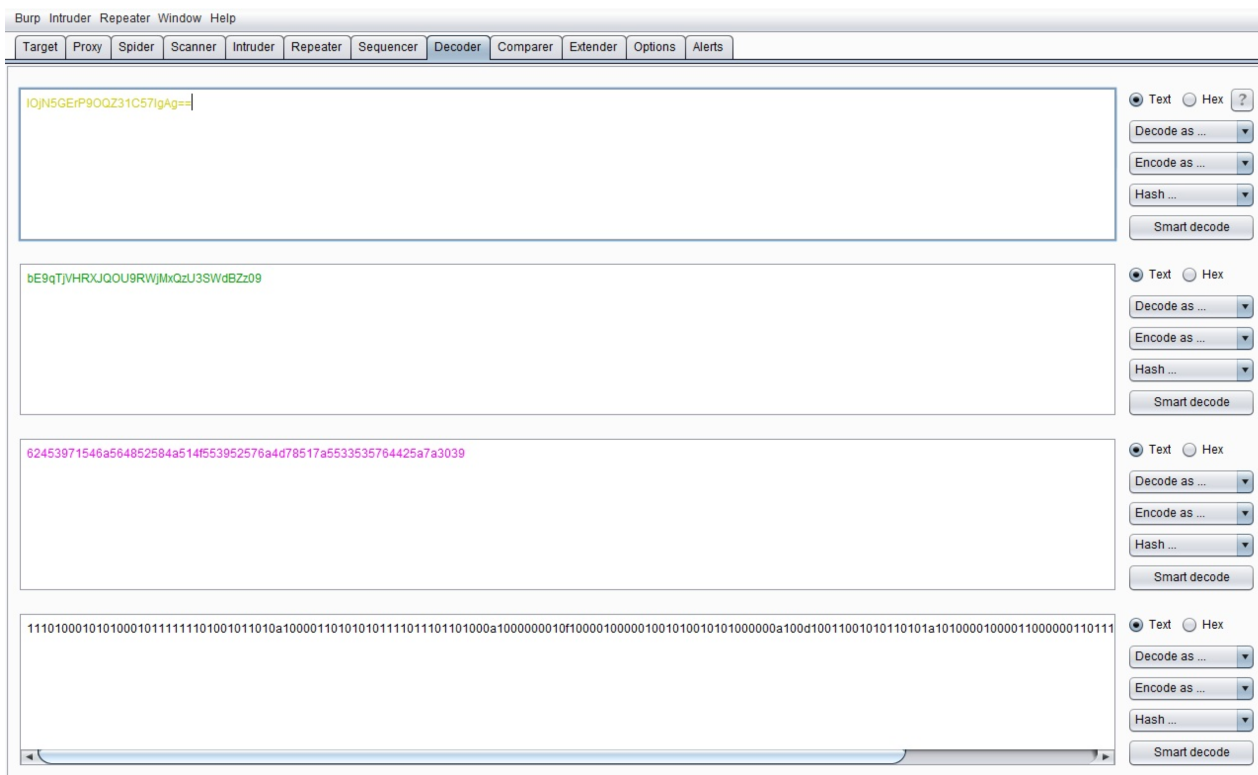
本章涉及诸多数学统计分析的知识，在表述或理解过程中由于知识水平的限制可能会存在错误，如果有问题的地方，欢迎发送邮件到 t0data@hotmail.com,先感谢您的批评指正。

第十一章 如何使用Burp Decoder

Burp Decoder的功能比较简单，作为Burp Suite中一款编码解码工具，它能对原始数据进行各种编码格式和散列的转换。其界面如下图，主要由输入域、输出域、编码解码选项三大部分组成。

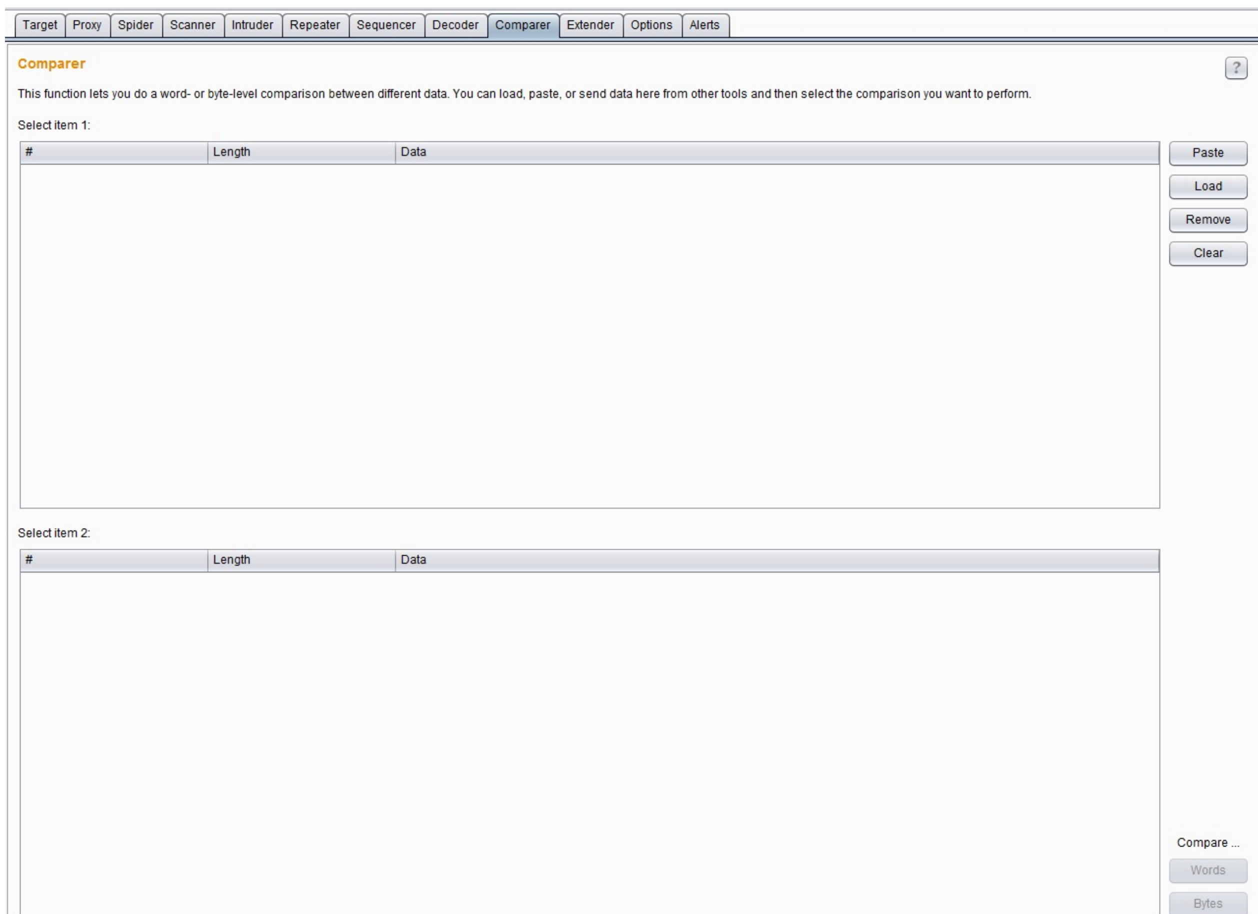


输入域即输入需要解码的原始数据，此处可以直接填写或粘贴，也可以通过其他Burp工具的上下文菜单中【Send to Decoder】；输出域即对输入域进行解码的结果显示出来。无论是输入域还是输出域都支持文本与Hex两种格式，其中编码解码选项中，由解码选项（Decode as)、编码选项（Encode as)、散列（Hash）三个构成。实际使用中，可以根据场景的需要进行设置。对于编码解码选项，目前支持URL、HTML、Base64、ASCII、16进制、8进制、2进制、GZIP共八种形式的格式转换，Hash散列支持SHA、SHA-224、SHA-256、SHA-384、SHA-512、MD2、MD5格式的转换，更重要的是，对于同一个数据，我们可以在Decoder的界面，进行多次编码解码的转换。如下图所示：

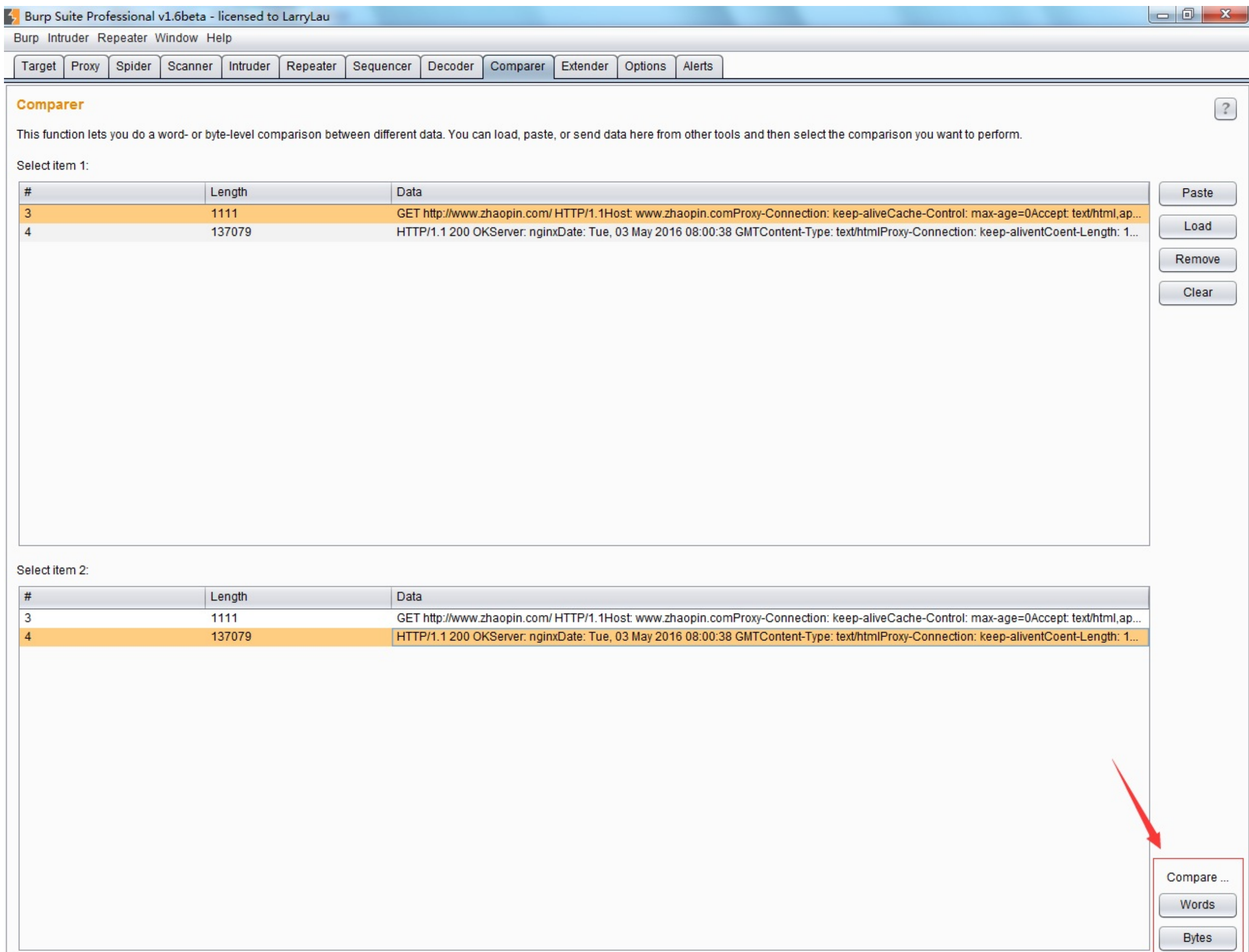


第十二章 如何使用Burp Comparer

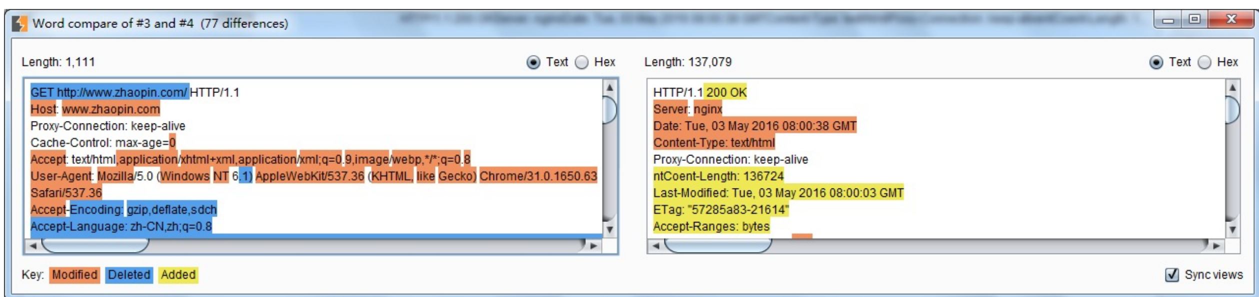
Burp Comparer在Burp Suite中主要提供一个可视化的差异比对功能，来对比分析两次数据之间的区别。使用中的场景可能是： 1.枚举用户名过程中，对比分析登陆成功和失败时，服务器端反馈结果的区别。 2.使用 Intruder 进行攻击时，对于不同的服务器端响应，可以很快的分析出两次响应的区别在哪里。 3.进行SQL注入的盲注测试时，比较两次响应消息的差异，判断响应结果与注入条件的关联关系。 其界面如下图：



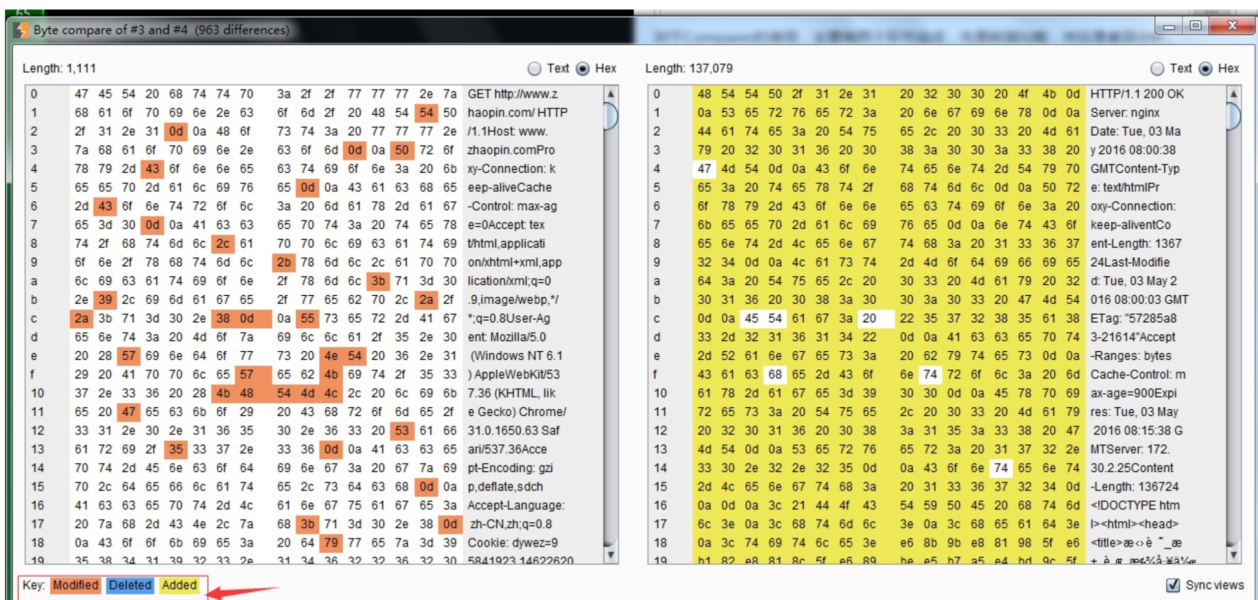
对于Comparer的使用，主要有两个环节组成，先是数据加载，然后是差异分析。 Comparer数据加载的方式常用的有：从其他Burp工具通过上下文菜单转发过来、直接粘贴、从文件加载三种方式。当加载完毕后，如果你选择了两次不同的请求或应答消息，则下发的比较按钮将被激活，可以选择文本比较或者字节比较。如下图：



如果点击了【words】或者【bytes】，则进入比对界面，页面自动通过背景颜色显示数据的差异。如下图：



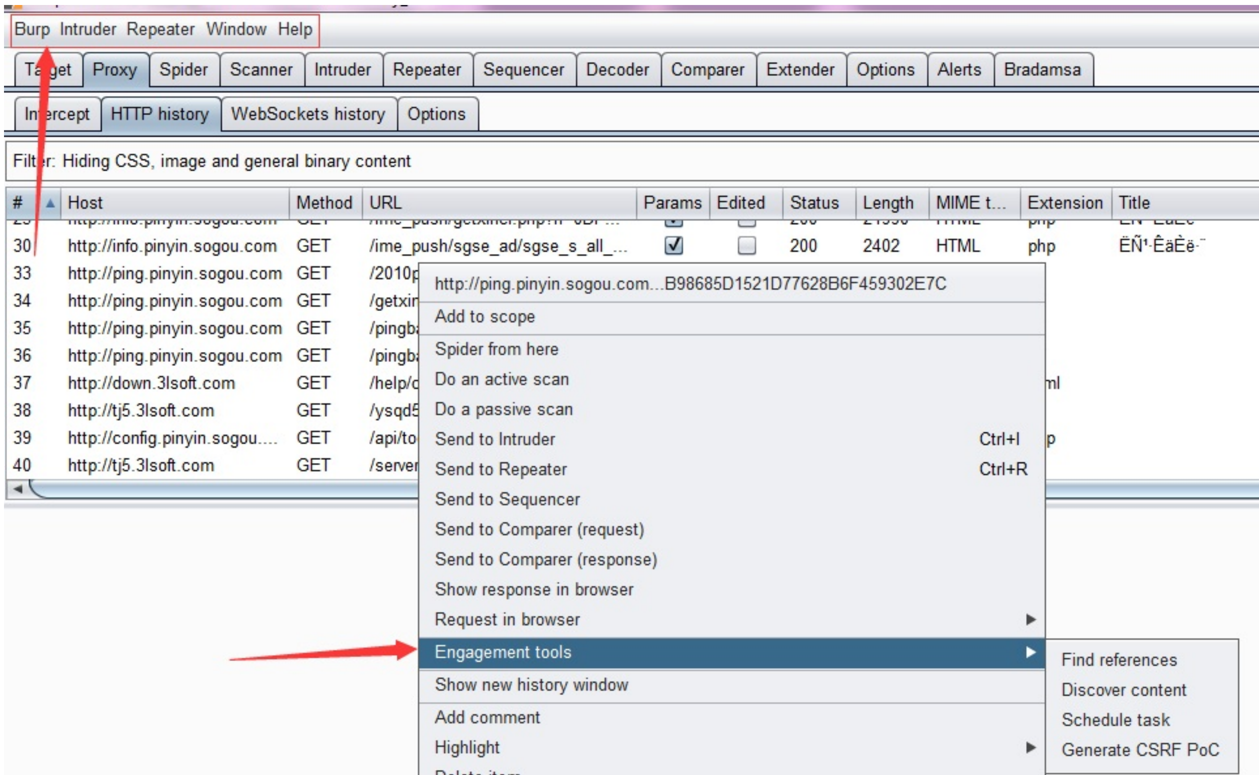
其中，文本比较（words）是指通过文本的方式，比如说以HTML的方式，比较两个数据的差异；而字节比较（bytes）是指通过16进制的形式，比较两次内容的差异。如下图,注意下发不同内容的颜色标注。



第十三章 数据查找和拓展功能的使用

通过第一部分十二个章节的学习，我们对BurpSuite的基本使用已经非常熟悉，从这一章开始，我们进入BurpSuite高级功能的使用。

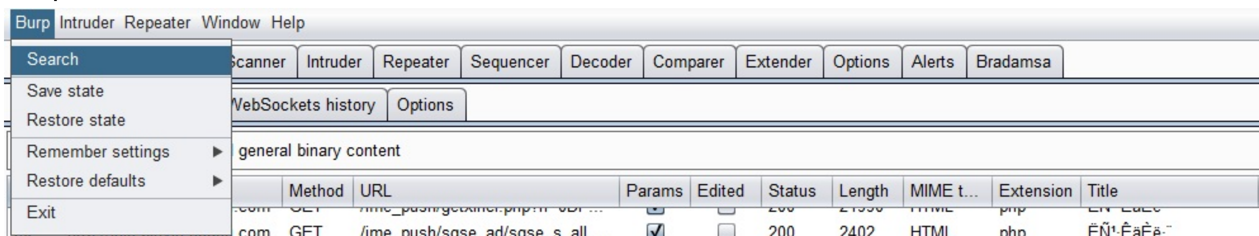
BurpSuite高级功能在界面布局上主要集中在两大块，一是菜单栏，另一个是右击菜单的Engagement tools。



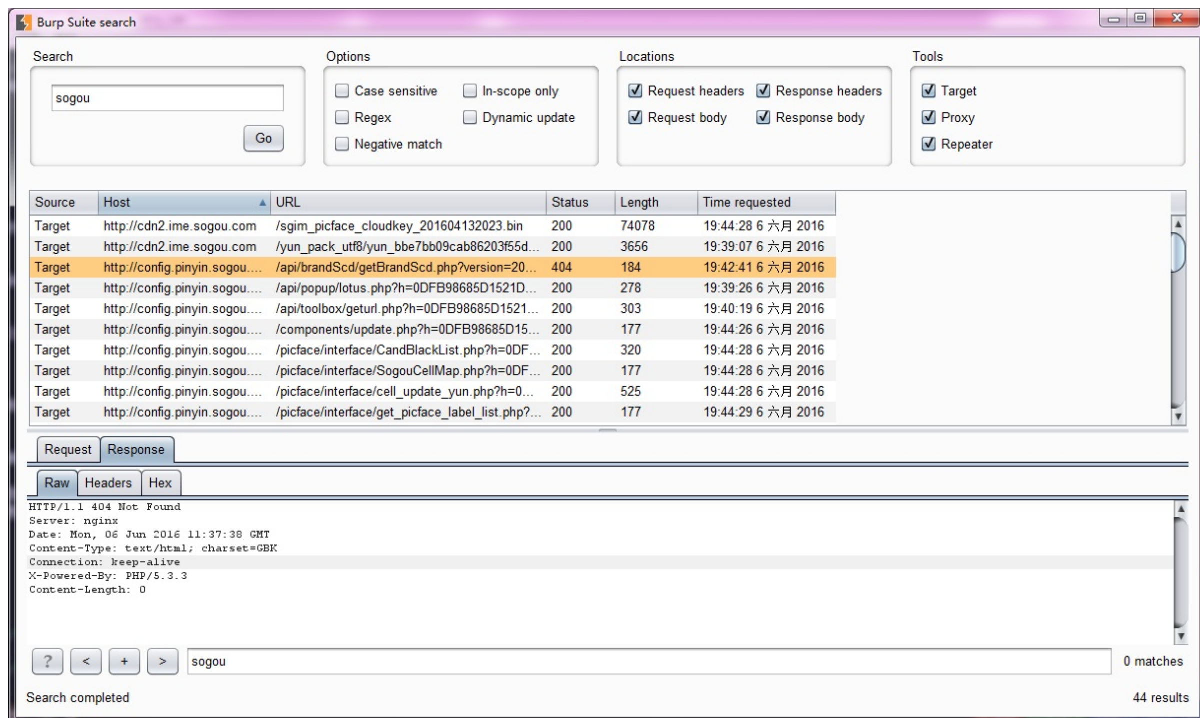
我们先来看看菜单栏，与日常使用相关的主要功能菜单是Burp、Intruder、Repeater.下面我们就逐一学习各个菜单的功能。

Burp

Burp 菜单下包含的数据查找（Search）、组件状态存储、组件状态恢复三部分。



- 数据查找（Search） 数据查找功能主要用来快速搜索Target、Proxy、Repeater三个组件中的请求和应答消息的内容，其界面如图：



默认情况下，当我们打开功能界面时，都是空的。如果我们在搜索框输入关键字，点击【Go】之后，下面的列表中将自动显示匹配到的所有消息。默认匹配时，是从HTTP消息中的Host、url、请求消息头，请求消息Body、应答消息头、应答消息Body中搜索匹配字段。 在整个Search面板中，有三大块设置项用于我们控制对数据的查询。

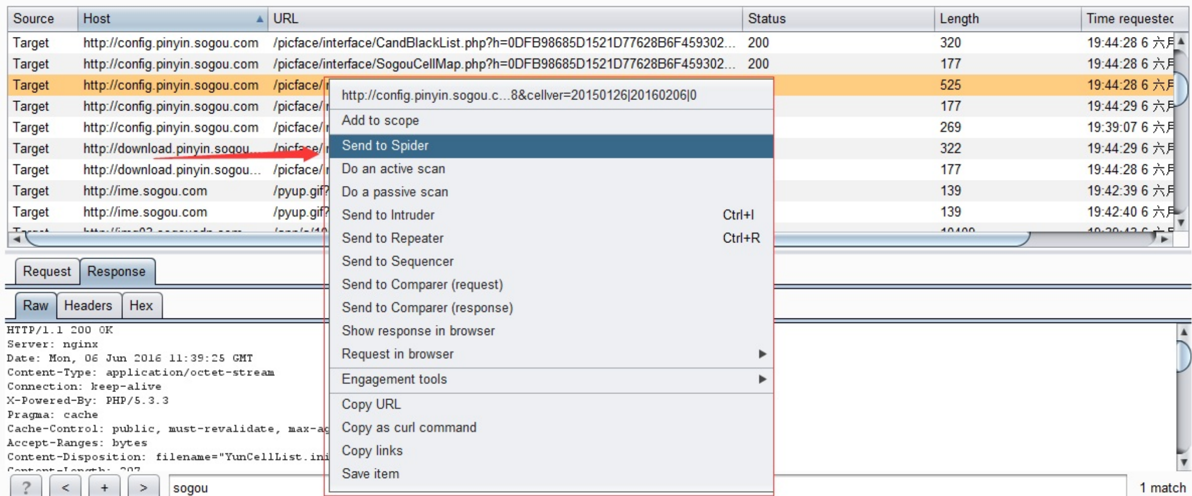
Options
Case sensitive
In-scope only
Regex
Dynamic update
Negative match

Locations
Request headers
Response headers
Request body
Response body

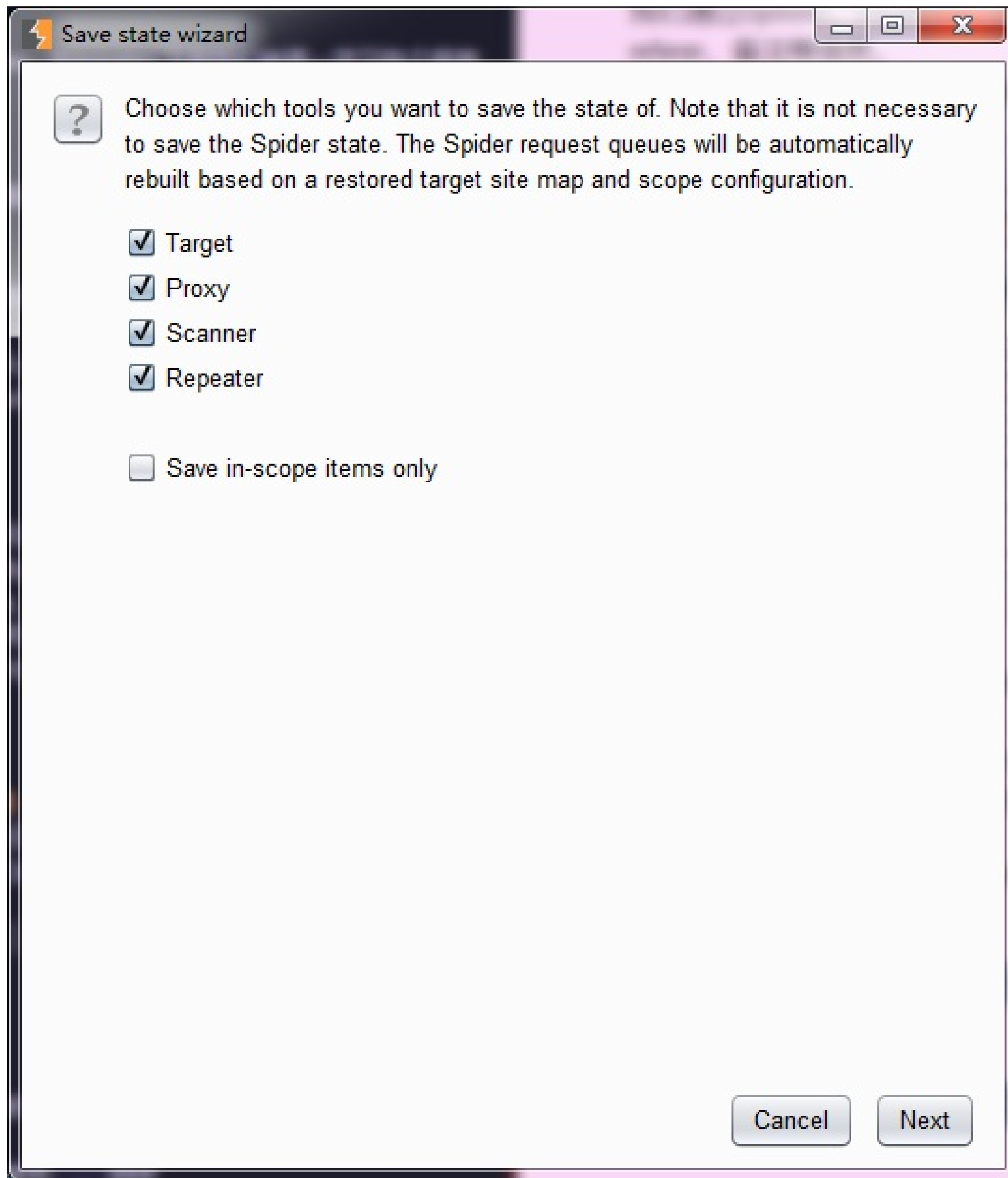
Tools
Target
Proxy
Repeater

Options主要控制关键字匹配的方式：大小写敏感、域内搜索、正则表达式匹配、动态更新、反向匹配 Locations主要用于控制关键字查找的范围：请求消息头、请求消息Body、应答消息头、应答消息Body

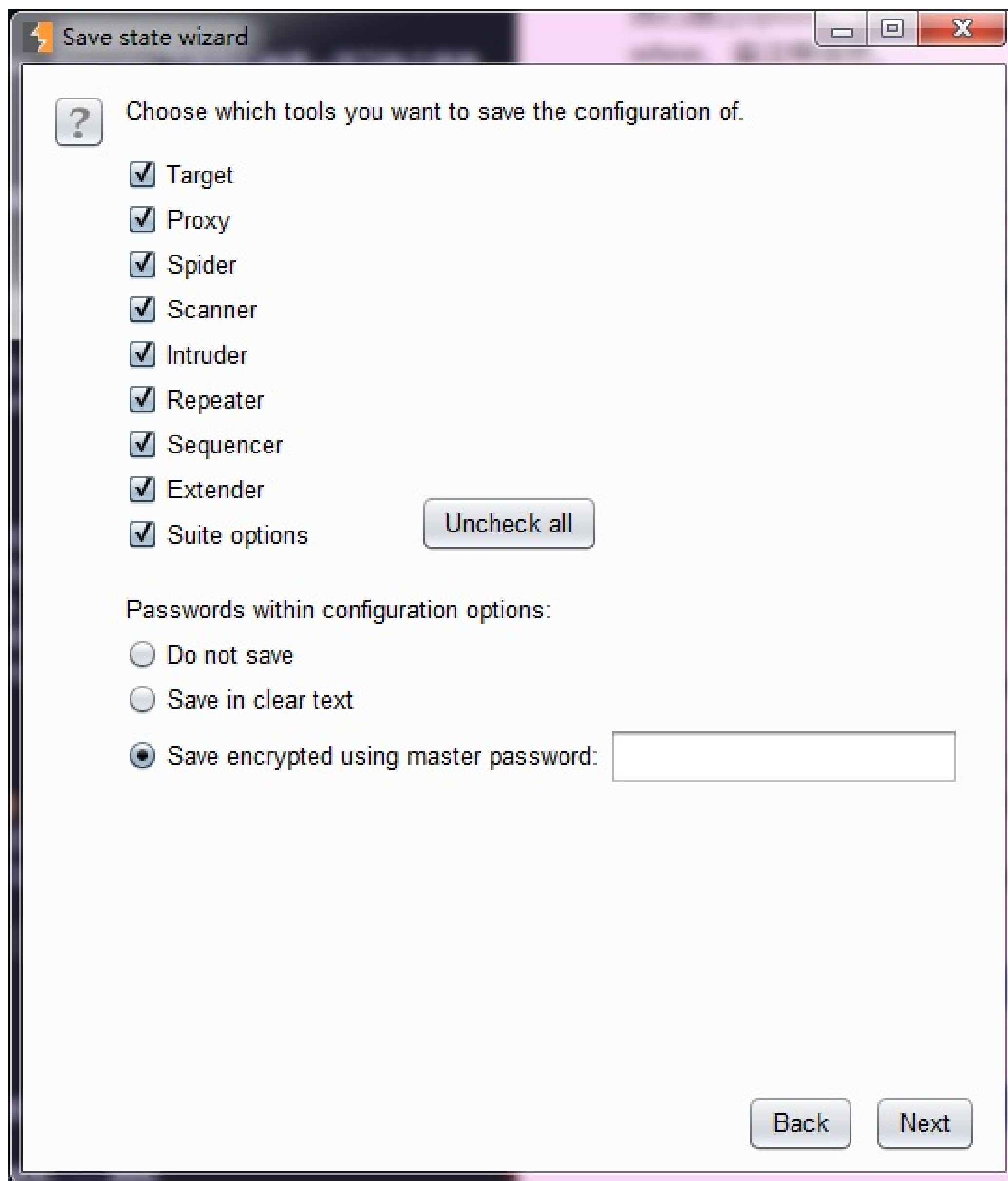
Tools主要用于控制关键字搜索的Burp工具组件的范围：Target、Proxy、Repeater 我们通过Options、Locations、Tools三者的组合，能准确的搜索我们关注的字符、脚本、referer、备注等信息。 当然，Search面板也集成了Burp的横向传递功能，当我们找到或发现关心的HTTP消息后，直接可传递到其他的工具组件中。



- 组件状态存储和恢复，与组件状态和恢复相关的子菜单比较多，分别是： **Save state** 保存当前Burp的状态，主要保存站点地图、Proxy历史日志、扫描的结果和正在扫描的队列、Repeater当前和历史记录、Suite其他工具组件的所有配置信息。当我们点击【Save state】时，Burp将会提示我们是否只保存Scope中的数据



同时，也会提示我们，是否对存储文件的存在的密码进行保存。你可以选择不保存、明文保存、使用主密码进行加密保存三种的任何一种。如果使用主密码加密，当你在恢复设置时，Burp将提示密码没有保存或者输入主密码。



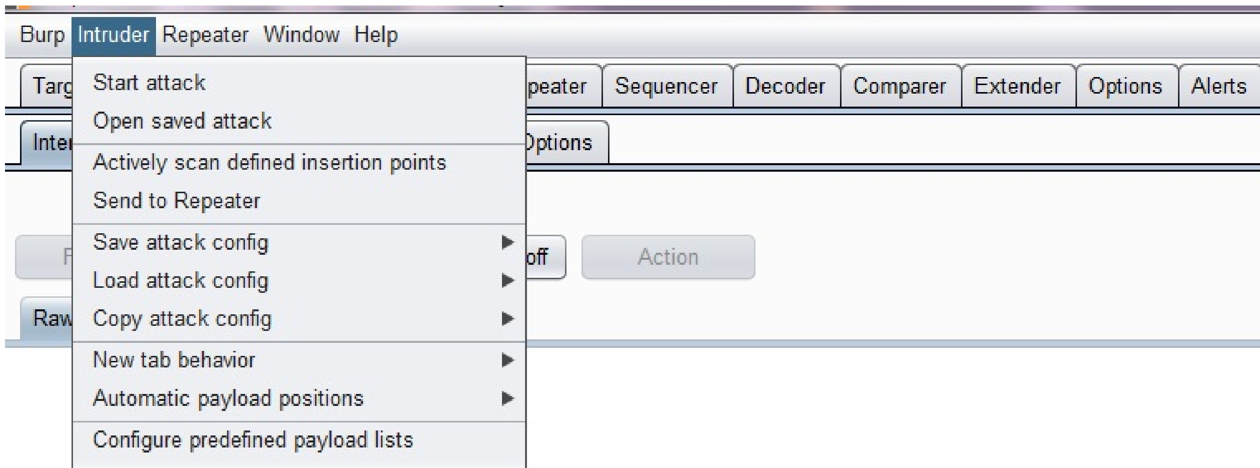
Restore state从之前的文件中恢复Burp之前保存的数据，与上面的**Save state**操作相对应。

使用组件状态存储和恢复的功能，能够帮助我们在渗透测试中带来极大的帮助。它主要体现在：

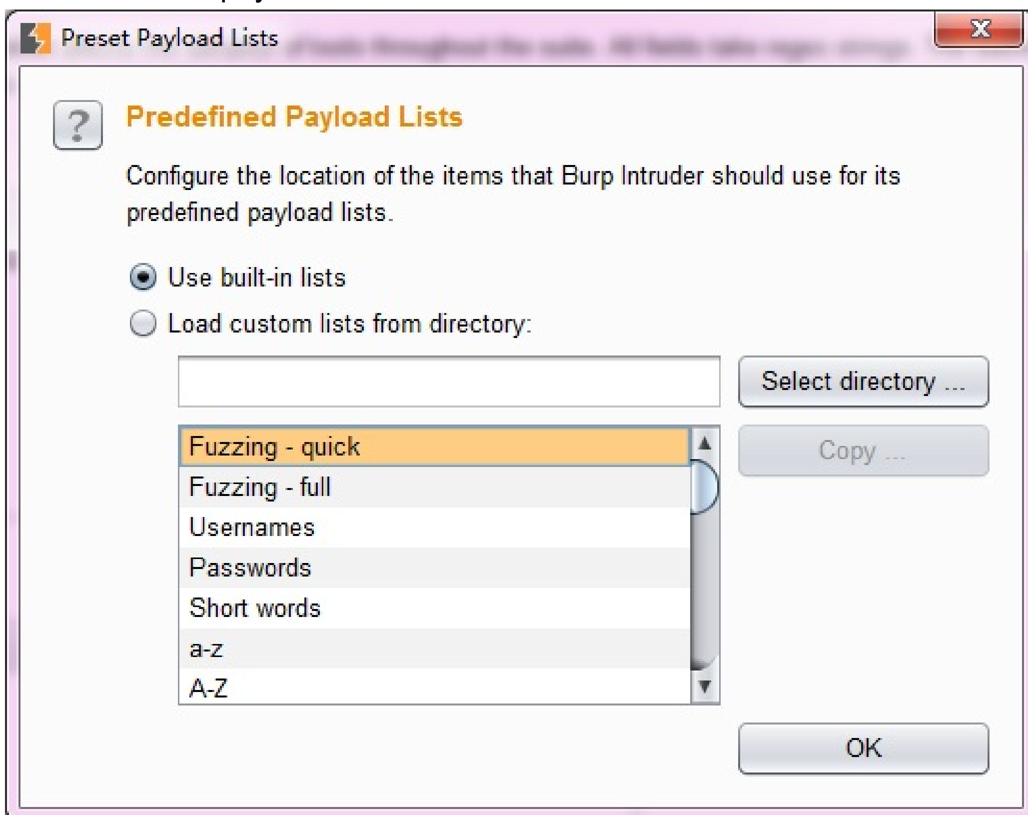
1. 保存你每一天的工作空间和进度以及问题的状态，以便于第二天查看。
2. 当系统发生故障或无法测试时，通过存储的Burp状态查看之前的问题和消息内容。
3. 通过归档的文件，你能跟踪已经修复的问题。
4. 通过所有的归档文件，对整个应用系统安全问题分布情况有总体的分析和评估。
5. 通过Burp状态文件作为模板，在团队间共享Burp配置和相关测试内容。

Intruder

Intruder菜单主要用于自动化攻击的相关配置。它的菜单和对应的功能如下：



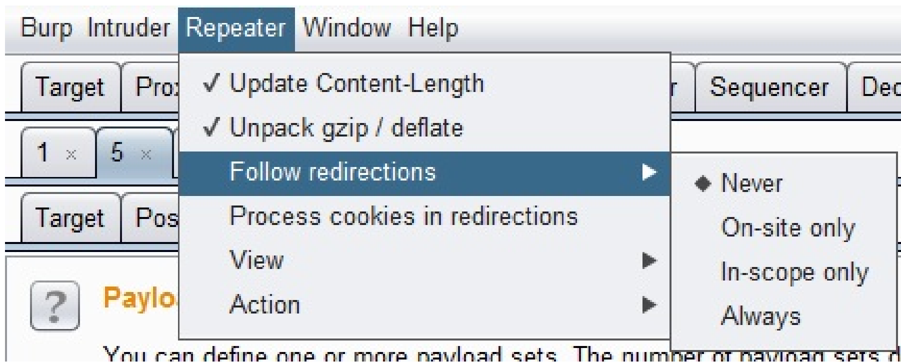
Start attack 开始发起攻击 **Open save attack** 重新加载之前保存的Intruder攻击文件 **Save attack config**、**Locd attack config**、**Copy attack config**，主要控制Intruder的攻击配置信息 **Automatic payload position**主要用于控制payload的使用方式：替换参数值或者追加参数值 **Configure predefined payload lists**用于控制Burp默认的payload字典值，当我们点击此菜单时，会弹出payload字典配置文件的界面，如下图所示：



我们可以选择一个payload子类型，对字典值进行修改。需要注意的事，这里选择的是payload文件存放的目录，当选择目录后，会自动加载目录下的payload文件。

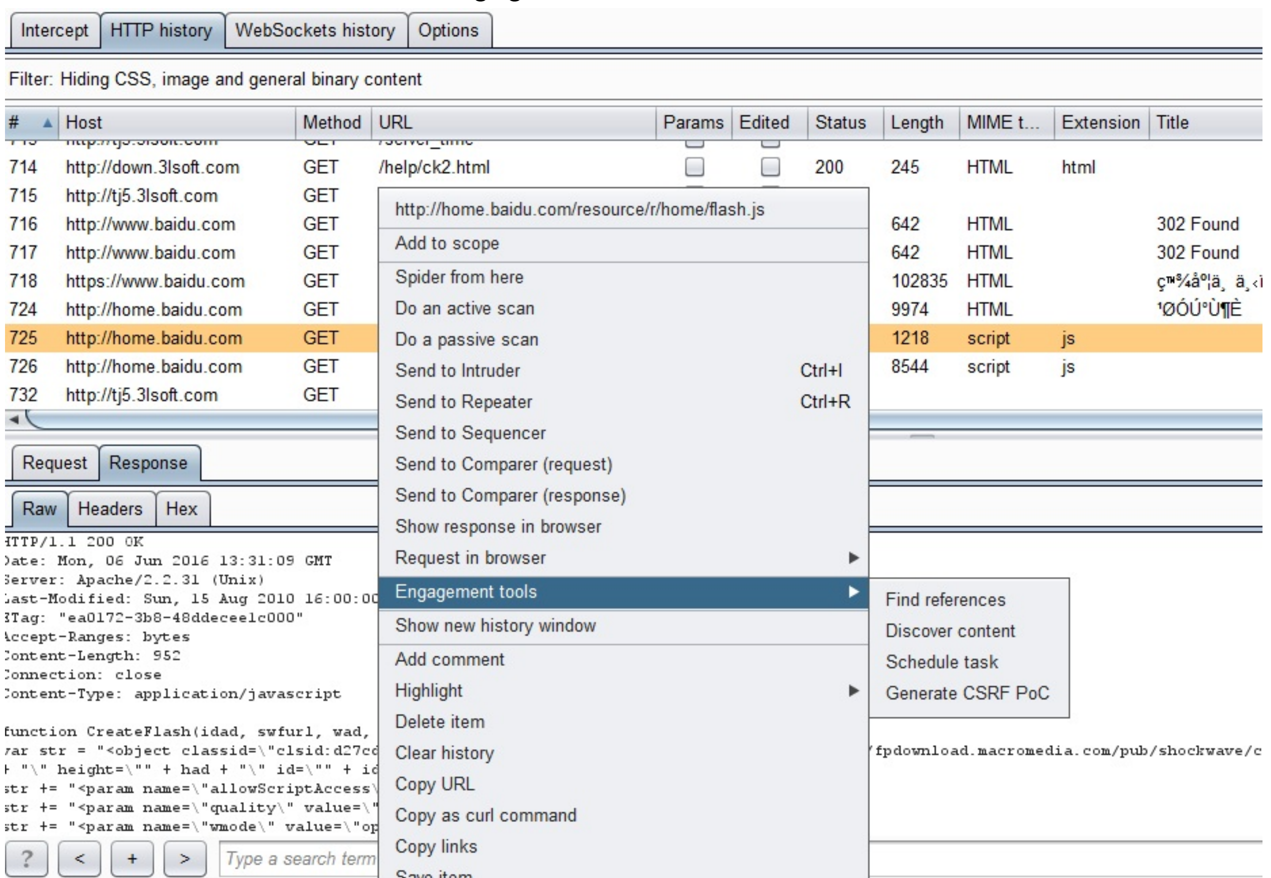
Repeater

Intruder菜单主要用于Repeater工具的控制，它的子菜单有：



Update Content-Length 当执行Repeater操作时，自动更新消息头中的Content-Length **Unpack gzip /deflate** 解压压缩文件 **Follow redirections** 跳转控制，可以选择从不跳转、同一站点内跳转、Scope内跳转、始终跳转四种的其中之一 **Process cookie in redirections** 跳转的同时是否处理Cookie **View**主要控制Repeater面板整个布局

熟悉完菜单栏之后，我们来看看Engagement tools。



从上图中我们知道，此功能位于右击菜单中，它包含**Find references**、**Discover content**、**Schedule task**、**Generate CSRF Poc**四个子菜单。

- **Find references**是指对选中的某条Http消息获取其referer信息

References to http://home.baidu.com/resource/r/home/flash.js

Source	Host	URL	Status	Length	Time requested
Target	http://home.baidu.com	/	200	9974	21:36:11 6 六月 2016

Request Response

Raw Headers Hex HTML Render

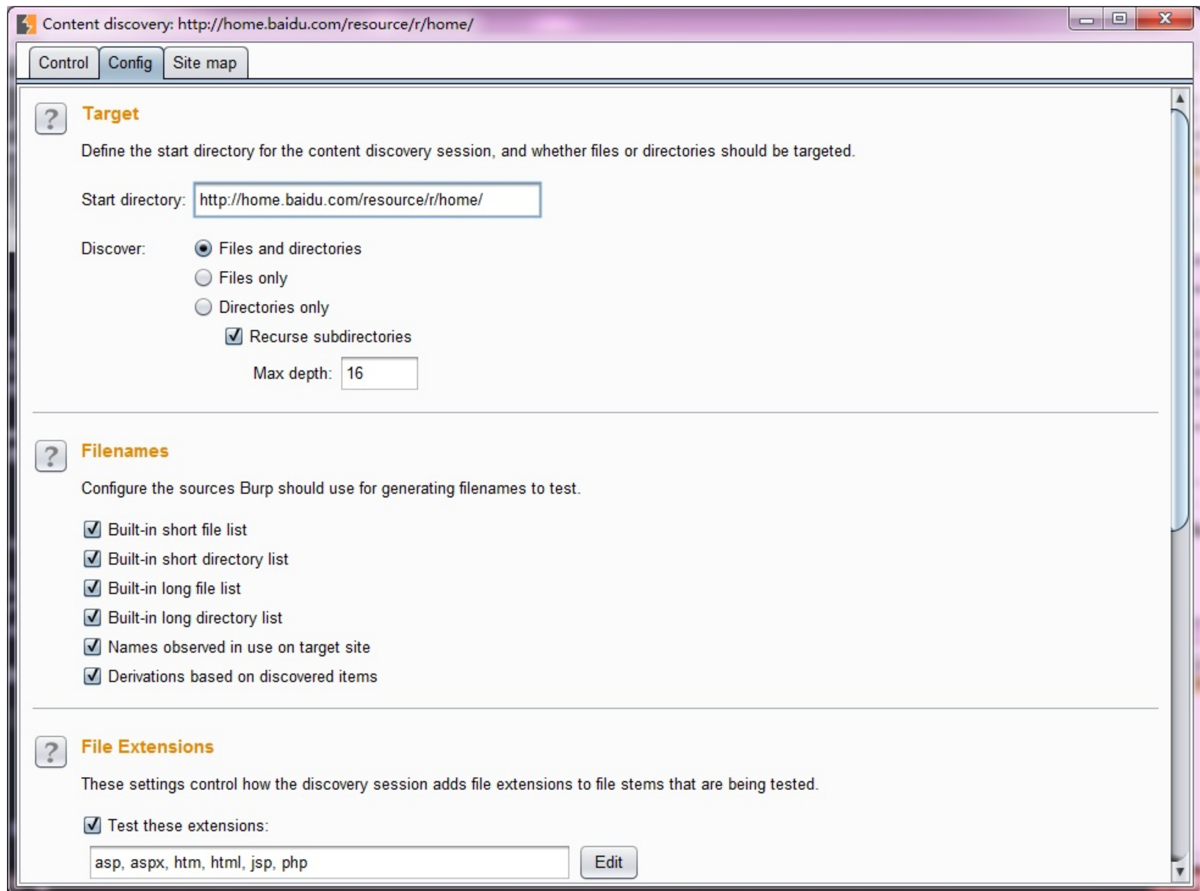
```
HTTP/1.1 200 OK
Date: Mon, 06 Jun 2016 13:31:08 GMT
Server: Apache/2.2.31 (Unix)
Last-Modified: Thu, 19 May 2016 03:00:47 GMT
ETag: "72186b-25f7-533292e4c95c0"
Accept-Ranges: bytes
Content-Length: 9719
Connection: close
Content-Type: text/html

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
```

? < + > Type a search term 3 highlights

Search completed 1 results

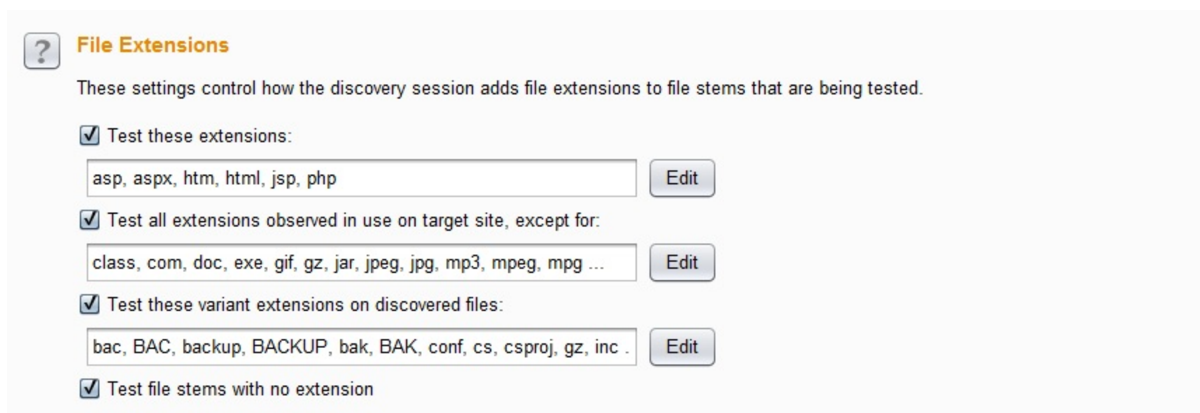
- **Discover content**是指对选中的某条Http消息，根据其url路径，进行目录枚举和文件枚举操作。当我们点击后，将弹出其配置界面。



- 其Discover选项有：挖掘文件和目录、仅仅挖掘文件、仅仅挖掘目录（递归遍历子目录，可指定其层级或深度）

- 挖掘的文件名（filenames）选项有：**Built-in short file list**内联的短文件列表 、**Built-in short directory list**内联的短目录列表、**Built-in long file list**内联的长文件列表、**Built-in long directory list**内联的长目录列表、**Names discovered in use on the target site**网站内发现的名称、**Derivations based on discovered item**基于已有名称进行猜测。

- 同时，如上图所示，我们也可以根据文件的拓展名对文件类型进行管理。



从上而下依次的含义是： **Test these extensions** 测试这些扩展名文件 **Test all extensions observed on target site**不测试这些扩展名文件，这个选项在我们不知道站点的大体情况下，我们可以去除那些我们熟悉的文件扩展名，然后去挖掘未知的扩展名

文件 **Test these variant extensions on discovered files**测试发现这些文件扩展名的变体，从图中我们可以看出，在测试备份文件的时候，这个选项会非常有用 **Test file stems with no extension**测试没有扩展名的文件

- 挖掘引擎配置选项有：

Discovery Engine

These settings control the engine used for making HTTP requests when discovering content.

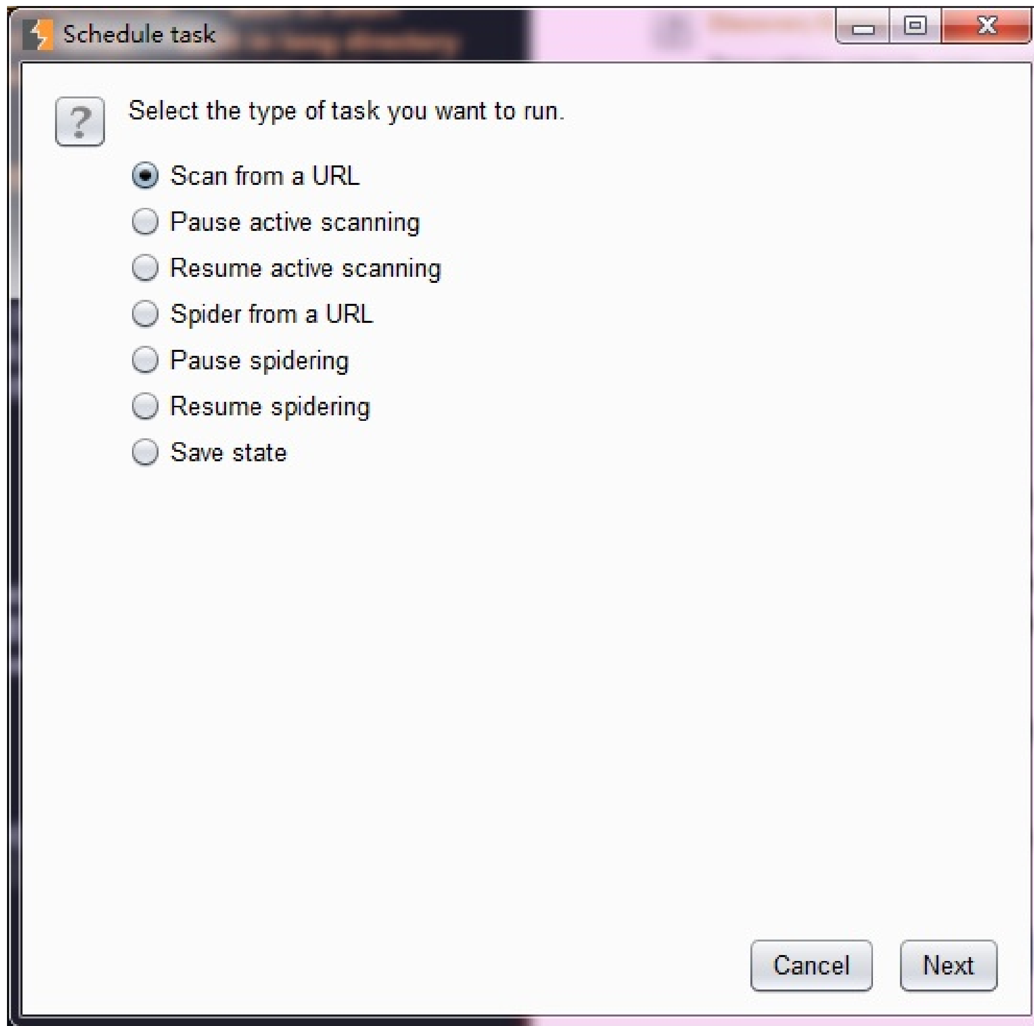
Case sensitivity: (●) Auto-detect ( ) Sensitive ( ) Insensitive

☑ Add discovered content to suite site map
☑ Copy content from suite site map
☑ Spider from discovered content

Number of discovery threads: 4

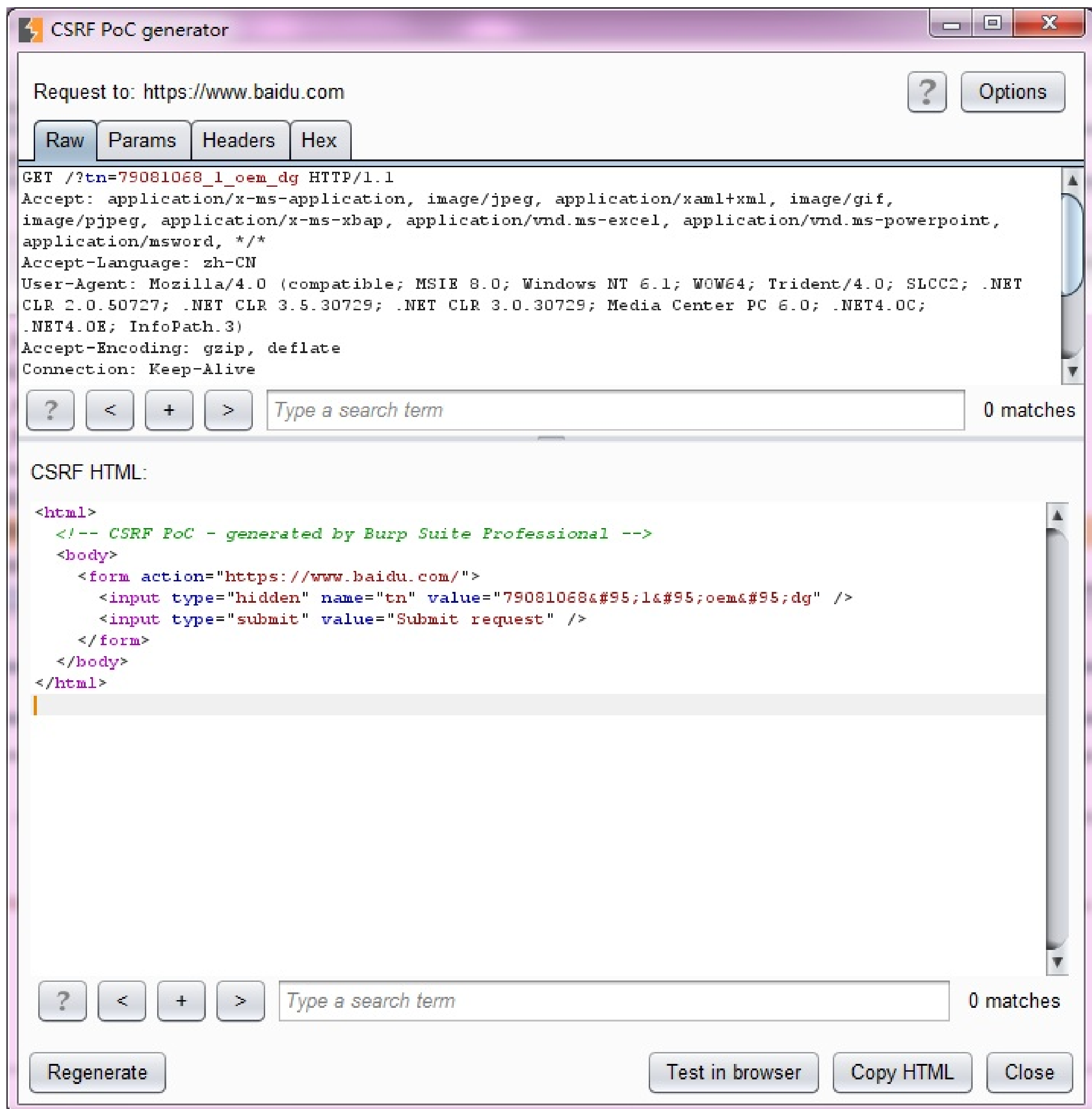
Number of spider threads: 2

主要有**Case sensitivity** 大小写敏感、**Add discovered content to suite site map** 添加挖掘结果到站点地图中、**Copy content from suite site map**复制Target站点地图到挖掘的站点地图中、**Spider from discovered content** 爬取挖掘到文件的内容、**Number of discovery threads**挖掘的线程并发数目、**Number of spider threads** 爬取的线程并发数目。

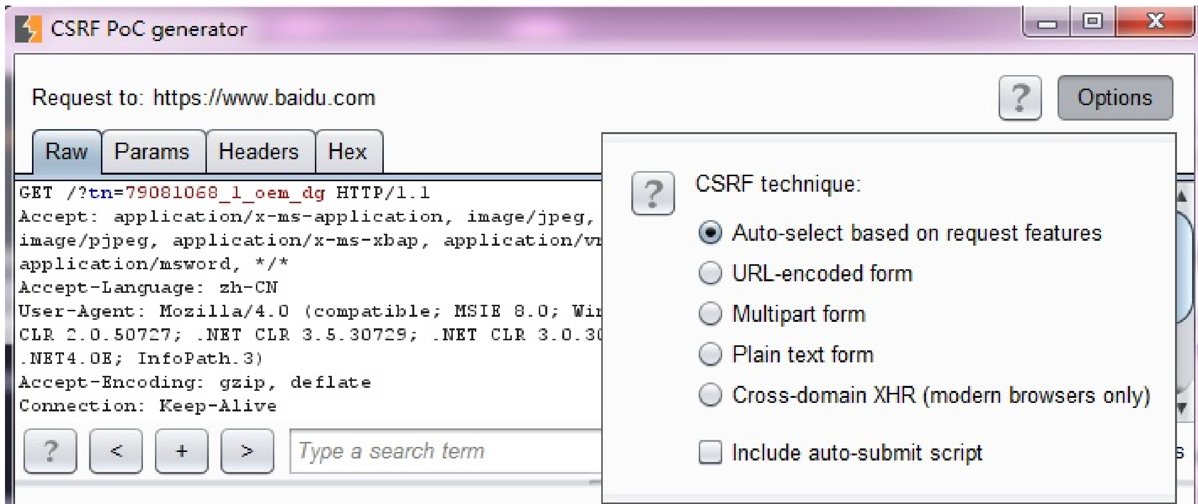
- **Schedule task**任务时间表 任务时间表的功能主要是把当前选中的url作为初始路径，然后进行多种任务的选择，进入任务时间表进行执行。

从图中我们可以看出，依据初始的url，我们可以做扫描、爬取、状态保存的相关操作。

- **Generate CSRF Poc**生成CSRF的POC 此功能的作用是，依据选中的http消息，自动生成CSRF的POC内容。当我们把POC的内容保存为HTML即可执行。

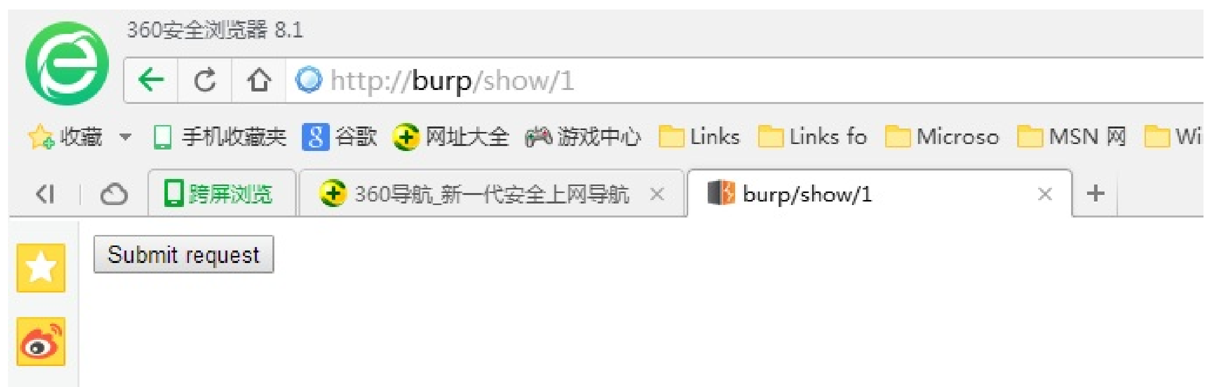


在生成POC时，我们可以对生成的参数进行设置，如图中右上角的【options】所示。



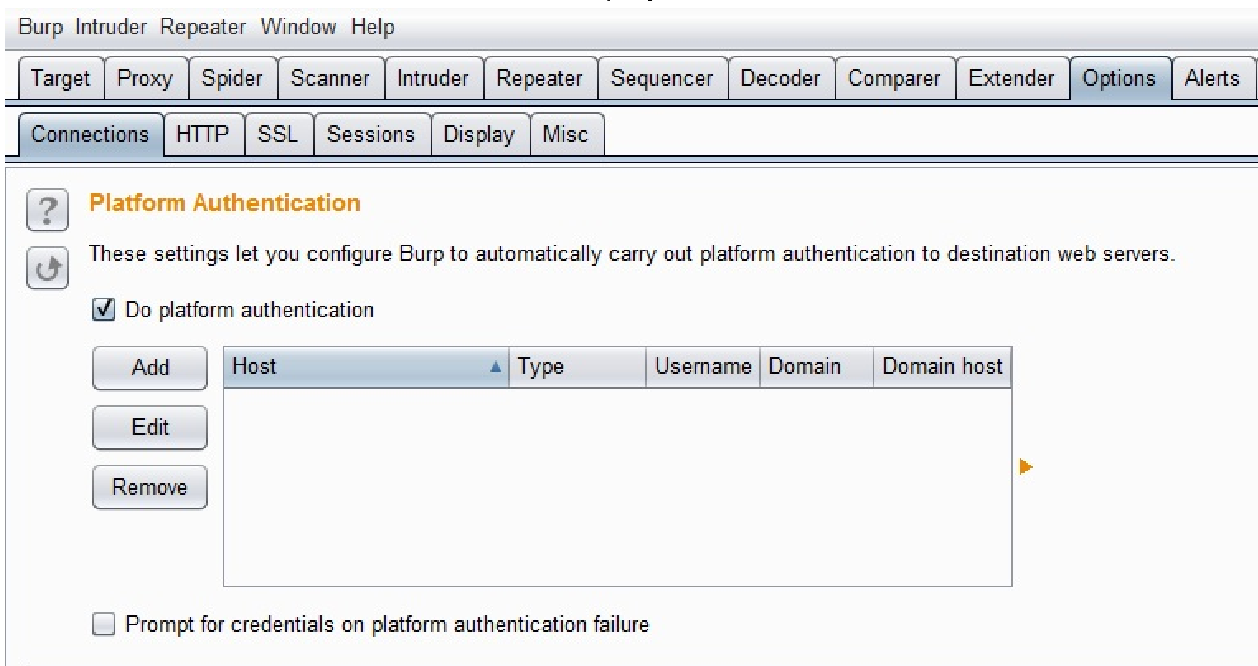
我们可以选择根据http特性自动生成、url编码的form表单、Mutipart类型的form表单、普通文本的form表单、跨域的异步请求以及自动提交，这些选项中一个或两个，当我们设置好之后，点击左下角的【Regenerate】重新生成即可。需要注意的是，Mutipart类型的form表单和普通文本的form表单的选择是由http消息中包含的content-type决定的。 如果修改了POC的生成设置，则需要点击左下角的【Regenerate】按钮，重新生成POC。当POC生成之后，你可以使用【CopyHTML】文本，放入html文件中进行浏览执行，也可以点击【Test in Brower】，在浏览器中直接预览执行，进行测试。




360安全浏览器 8.1
http://burp/show/1
收藏
手机收藏夹
谷歌
网址大全
游戏中心
Links
Links fo
Microso
MSN 网
跨屏浏览
360导航_新一代安全上网导航
burp/show/1
Submit request

第十四章 BurpSuite全局参数设置和使用

在Burp Suite中，存在一些粗粒度的设置，这些设置选项，一旦设置了将会对Burp Suite的整体产生效果，这就是Burp Suite中Options面板。当我们打开Options面板即可看到，它是由Connections、HTTP、SSL、Sessions、Display、Misc六个选项卡组成。



本章的内容主要包括：

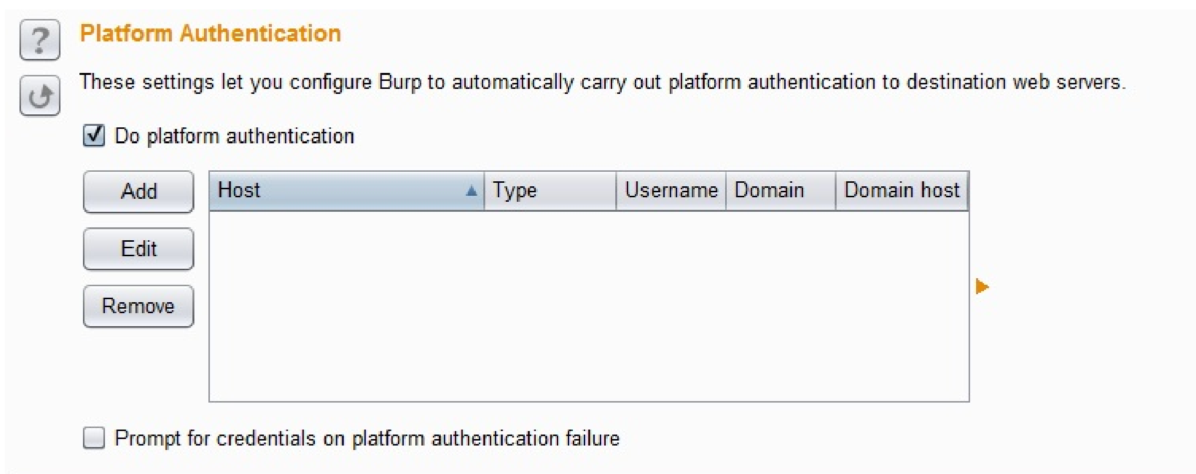
- Burp 网络连接设置（Connections）
- HTTP应答消息处理设置（HTTP）
- SSL连接和加密设置（SSL）
- 会话设置（Sessions）
- 显示设置（Display）
- 其它工具设置（Misc）

下面我们就依次来看看每一个选项卡包含哪些详细的功能设置。

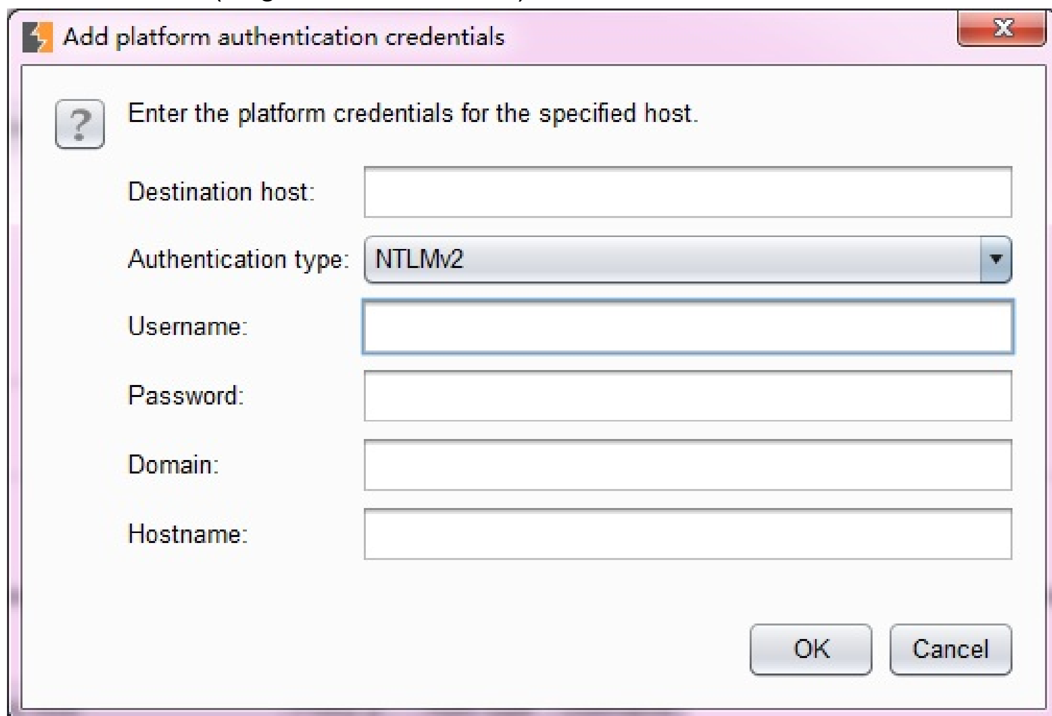
Burp 网络连接设置（Connections）

Connections选项卡主要用来控制Burp如何来处理平台认证、上游代理服务器、Socks代理、超时设置、主机名或域名解析以及Scope之外的请求六个方面的相关配置。当我们打开Connections选项卡，从上往下拖动，首先看到的设置将是平台身份认证（Platform Authentication）。

- 平台身份认证（Platform Authentication）



这些设置允许你配置Burp自动执行到目标Web服务器的平台身份验证，不同的主机可以配置不同的认证方式和证书。目前支持的身份验证类型有：BASIC，NTLMv1，NTLMv2和“摘要”式认证(Digest authentication)。其设置界面截图如下：

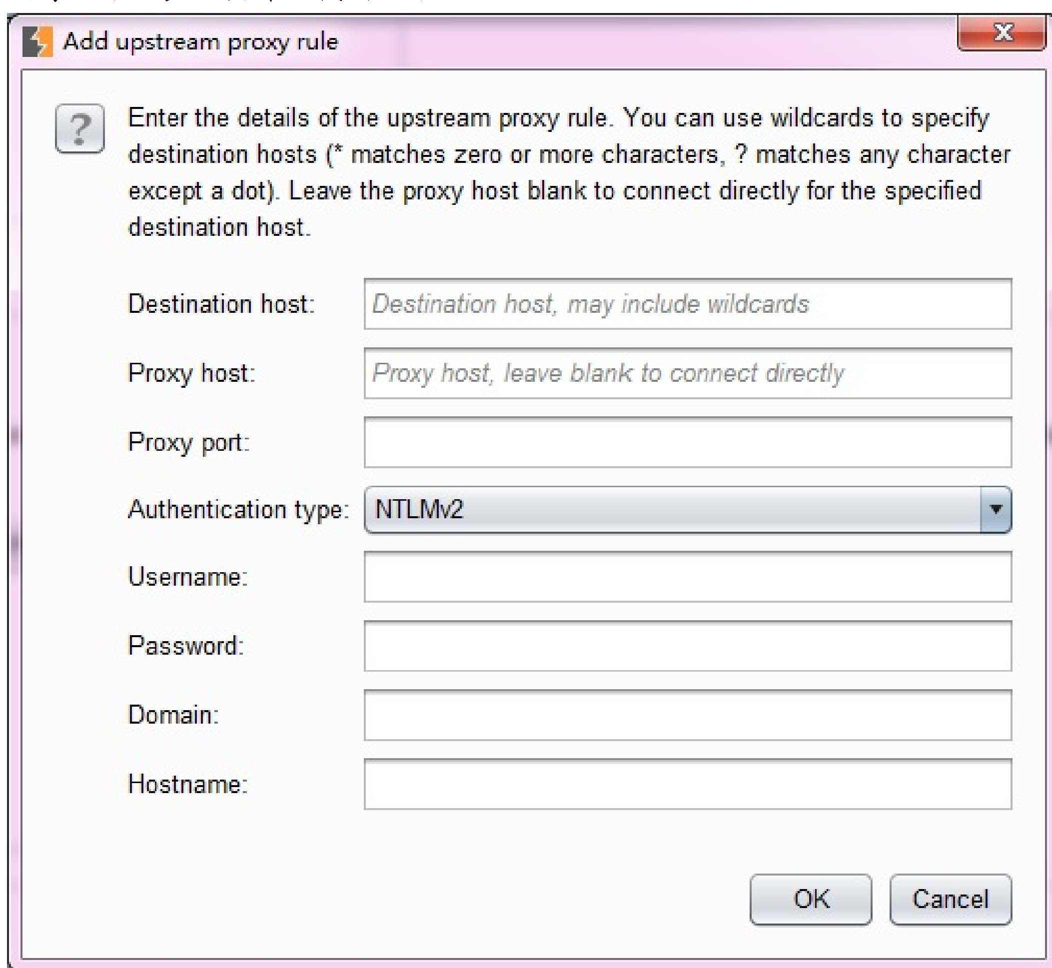


其中域名和主机名字段只用于NTLMv1，NTLMv2身份验证。 在平台身份认证（Platform Authentication）设置的最下方有一个Checkbox选项（Prompt for credentials on platform authentication failure），如果此项选中，则表示当遇到身份验证失败时，Burp会显示一个交互式的弹窗，提示验证失败的信息。

- 上游代理服务器（Upstream Proxy Servers）



这些设置主要是控制Burp是否会发送请求到上游代理服务器，或直转向目标Web服务器。从代理服务器配置的图中我们可以看出，这是一个列表，那就表明我们可以配置多个匹配规则。当我们配置了多个规则时，可以针对不同的目标主机或主机组指定不同的代理服务器设置。这些规则将按照顺序，并将与目标Web服务器相匹配的第一个规则作为生效规则。如果列表没有规则匹配，Burp默认采取直连、非代理的方式进行连接。 针对每一个配置，其界面截图如下：



我们可以使用在目标主机输入框中采用正则表达式，使用通配符（*零个或多个字符匹配？与任何字符相匹配，除了一个点）。来指定将所有请求发送到一个代理服务器。而对于配置的每个上游代理服务器，我们可以根据需要指定认证方式和认证凭据。它支持的身份认证类型有：BASIC，NTLMv1，NTLMv2和“摘要式”身份验证。同样，域名和主机名字段只用于NTLM身份认证。 当我们每配置完成一条匹配规则之后，它将出现在上游代理服务器的列表中，我们可以在列表中对其进行内容的编辑和上下顺序的调整。

- Socks代理

SOCKS Proxy

These settings let you configure Burp to use a SOCKS proxy. This setting is applied at the TCP level, and all outbound requests will be sent vi
requests to upstream proxies will be sent via the SOCKS proxy configured here.

Use SOCKS proxy

SOCKS proxy host:

SOCKS proxy port:

Username:

Password:

Do DNS lookups over SOCKS proxy

这些设置允许我们配置Burp使用SOCKS代理的方式进行所有传出的通信，但此设置只在TCP层生效，所有出站请求将通过这个代理发送。如果我们同时设置了已游HTTP代理服务器配置的规则，则请求上游代理将通过这里配置的SOCKS代理发送。其请求的匹配路径依次是：本地-->上游代理-->SOCKS代理。 在使用SOCKS代理时，我们需要勾选【Use SOCKS proxy】，并提供代理的ip或者主机名、端口、认证的用户名和口令（如上图所示）。 如果我们勾选了【Do DNS lookups over SOCKS proxy】，则进行域名解析时，将通过SOCKS代理去查询，而不会使用本地缓存。

- 超时设置（Timeouts）

Timeouts

These settings specify the timeouts to be used for various network tasks. Values are in seconds.

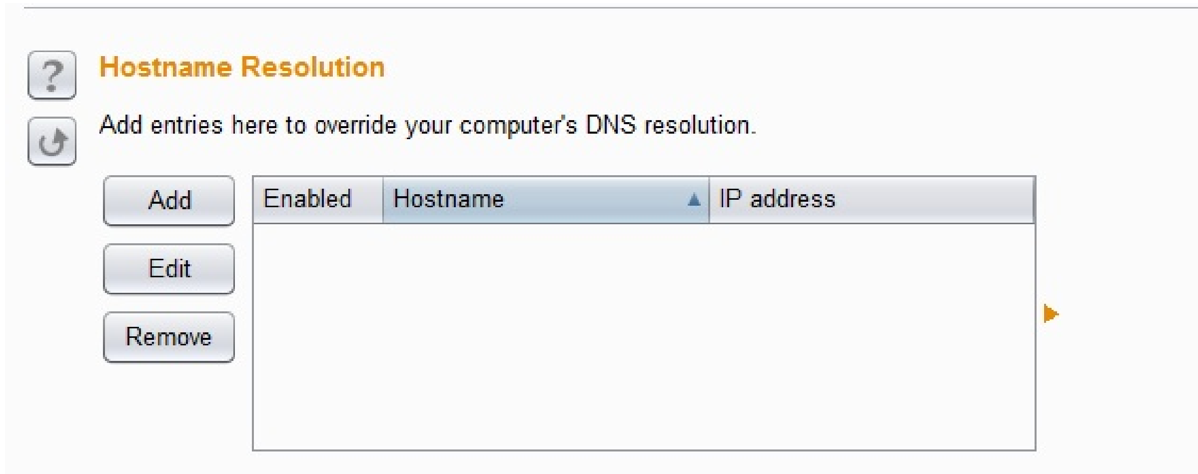
Normal:	120
Open-ended responses:	10
Domain name resolution:	300
Failed domain name resolution:	60

这些设置主要用于指定Burp各种网络任务的超时。我们可以对以下超时项进行设置：

- 正常（Normal ） - 此设置用于大多数网络通信，并确定Burp怎样放弃请求和记录已发生超时前等待。
- 开放式应答（Open-ended responses） - 该设置只用在一个响应正在处理不包含内容长度或传输编码HTTP标头。在这种情况下，Burp确定传输已经完成之前等待指定的时间间隔。 3.域名解析（Domain name resolution ） - 此设置确定Burp如何重新进行成功的域名查找，如果目标主机地址频繁变化时需要设定为一个适当的低的值。
- 失败的域名解析（Failed domain name resolution ） - 此设置确定Burp多久会重新尝试不成功的域名查找。

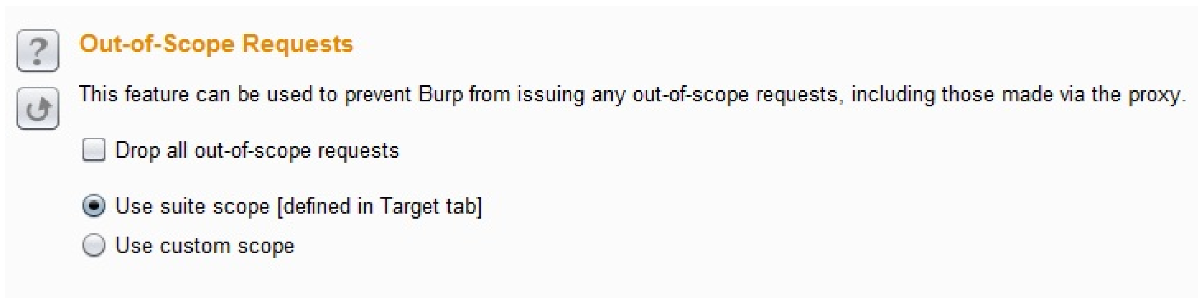
以上的选项设置的值都是以秒为时间单位，如果一个选项留空，那么表示Burp永远不会超时。

- 主机名或域名解析



此项配置比较简单，通过这些设置，我们可以指定主机名映射到IP地址，来覆盖本地计算机提供的DNS解析。每个主机名解析规则需要指定主机名，并与主机名相关联的IP地址。同时，每一个规则可以单独启用或禁用来控制其是否生效。当我们在渗透测试中，如果使用了隐形代理来测试富客户端组件，此功能可以确保请求正确转发。

- Scope之外的请求



这一特性可用于防止Burp发送任何超出Target面板中设置的Scope范围之外的请求，当我们需要保证没有请求到不在Scope范围内为它是有用的。例如，如果我们勾选了【Drop all out-of-scope requests】，即使你的浏览器使得超出范围的目标请求，这些请求也会被Burp被丢弃。 当然，我们可以启用此功能为当前的目标范围，如图，选中【Use suite scope】。或者，可以使用URL匹配规则定义自定义范围，选中【Use custom scope】。当我们选中【Use custom scope】时，界面将会显示其相关URL匹配规则的详细设置。如下图：

Use custom scope

Include in scope

Add
Edit
Remove
Paste URL
Load ...

Enabled	Protocol	Host / IP range	Port	File

Exclude from scope

Add
Edit
Remove
Paste URL
Load ...

Enabled	Protocol	Host / IP range	Port	File
☑	Any			logout
☑	Any			logoff
☑	Any			exit
☑	Any			signout

和Target Scope配置类似，它也分包含域和排除域，因其配置方式与Scope一致，此处就不在赘述。如果配置中有不明白的地方，请参数Target Scope配置章节

Session设置

会话处理规则（**Session Handling Rules**）

Connections | HTTP | SSL | Sessions | Display | Misc

Session Handling Rules

You can define session handling rules to make Burp perform specific actions when making HTTP requests. Each rule has a defined scope (for particular tools, URLs or parameters), and can perform actions such as adding session cookies, logging in to the application, or checking session validity. Before each request is issued, Burp applies in sequence each of the rules that are in-scope for the request.

Add
Edit
Remove
Duplicate
Up
Down

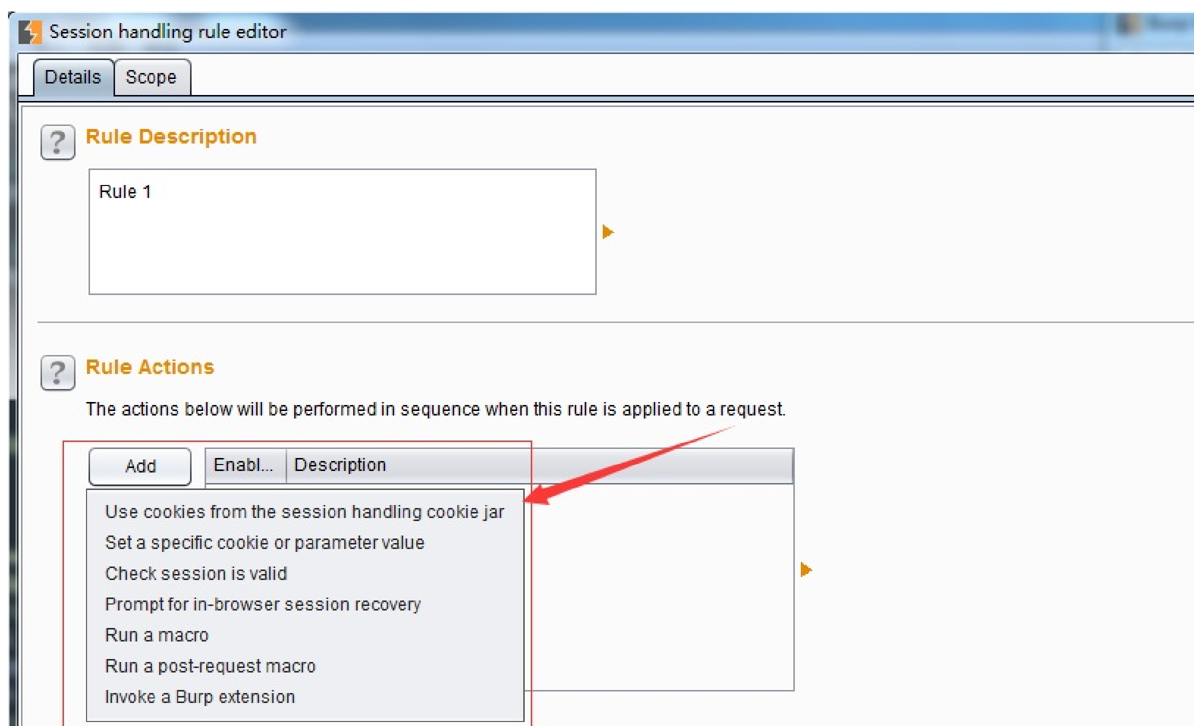
Enabled	Description	Tools
☑	Use cookies from Burp's cookie jar	Spider and Scanner

To monitor or troubleshoot the behavior of your session handling rules, you can use the sessions tracer to view in detail the results of processing each rule.

Open sessions tracer

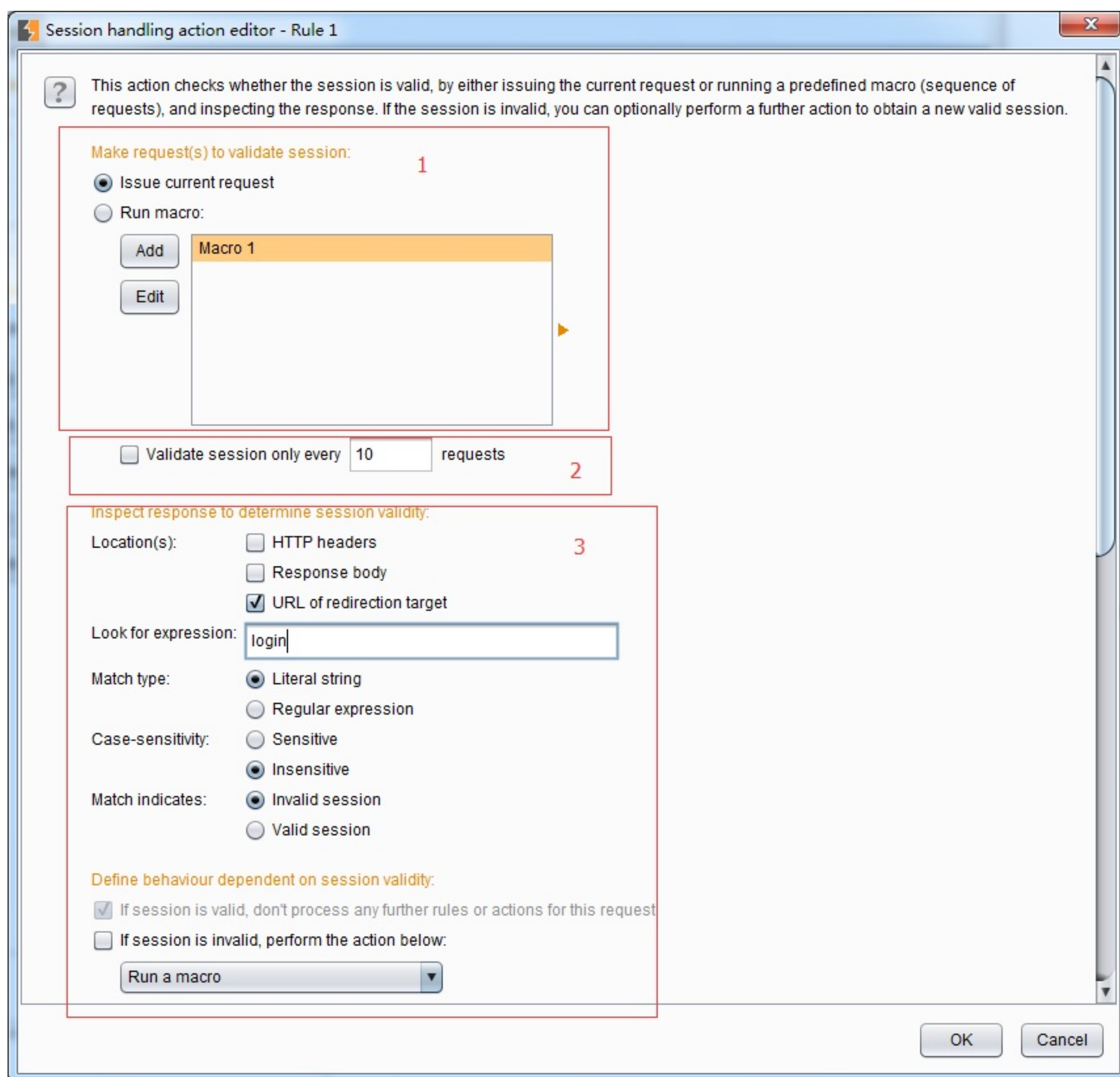
如上图所示，Burp允许你自定义会话处理规则的列表，这能让我们细粒度地控制Burp如何处理应用程序的会话处理机制和相关功能。对于处理规则，Burp中规则的构成包括范围（规则适用于）和动作（规则做什么），当我们点击【Add】按钮，弹出的规则配置界面如下图所示，其中**Details**和**Scope**两个面板的设置分别对于于上文的动作和范围。

- 动作（Rules Action）

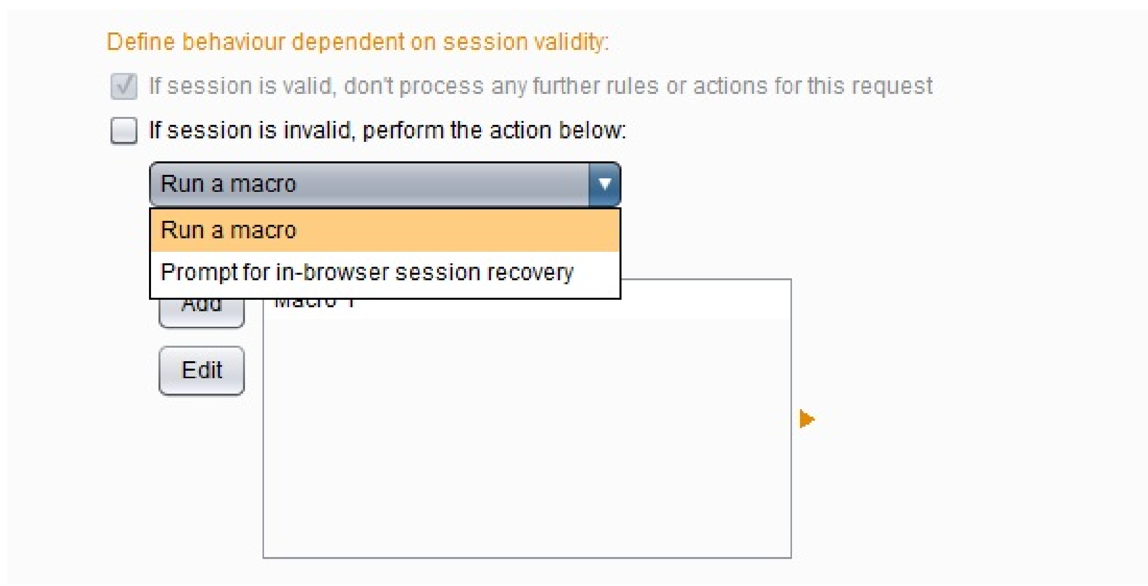


每个规则可以执行一个或多个操作，例如： 从Burp的cookie jar中更新cookies、验证当前会话、 运行宏（预定义的请求序列）等等。通过创建具有不同范围和操作的多个规则，您可以定义Burp将应用于不同应用程序和函数的行为的层次结构。例如，在特定测试中，您可以定义以下规则：对于所有请求，从Burp的cookie jar添加cookie；对于对特定域的请求，请验证与该应用程序的当前会话是否仍处于活动状态，如果没有，请运行宏以重新登录到应用程序，然后使用生成的会话令牌更新cookie jar；对于包含csrftoken参数的特定URL的请求，首先运行宏以获取有效的csrftoken值，并在发出请求时使用此值。 在Details面板中，Burp已经预制了七类规则动作，他们分别是：

1. **Use Cookies From the Session Handling Cookie Jar** 这个配置的动作是通过Burp的Cookie.jar用来更新请求的cookie信息，当然，你可以设置更新全部的cookie还是有选择性的更新。
2. **Set a Specific Cookie or Parameter Value** 这个配置的动作是指定cookie或者某个参数的值，如果没有设置的话，则在会话中添加此参数或者cookie。
3. **Check Session Is Valid** 此动作是检查当前会话是否有效，如果无效，则可选择地执行下一步的动作以获得新的有效会话。或者，我们可以将Burp配置为仅每X个请求验证会话，这有助于避免在应用程序发出多余的请求（==下图中2部分所示==）。 为了确定当前会话的有效性，Burp通常会发出一个或多个请求。这些请求可能是（==下图中1部分所示==）：a）当前的会话请求 b）执行宏脚本

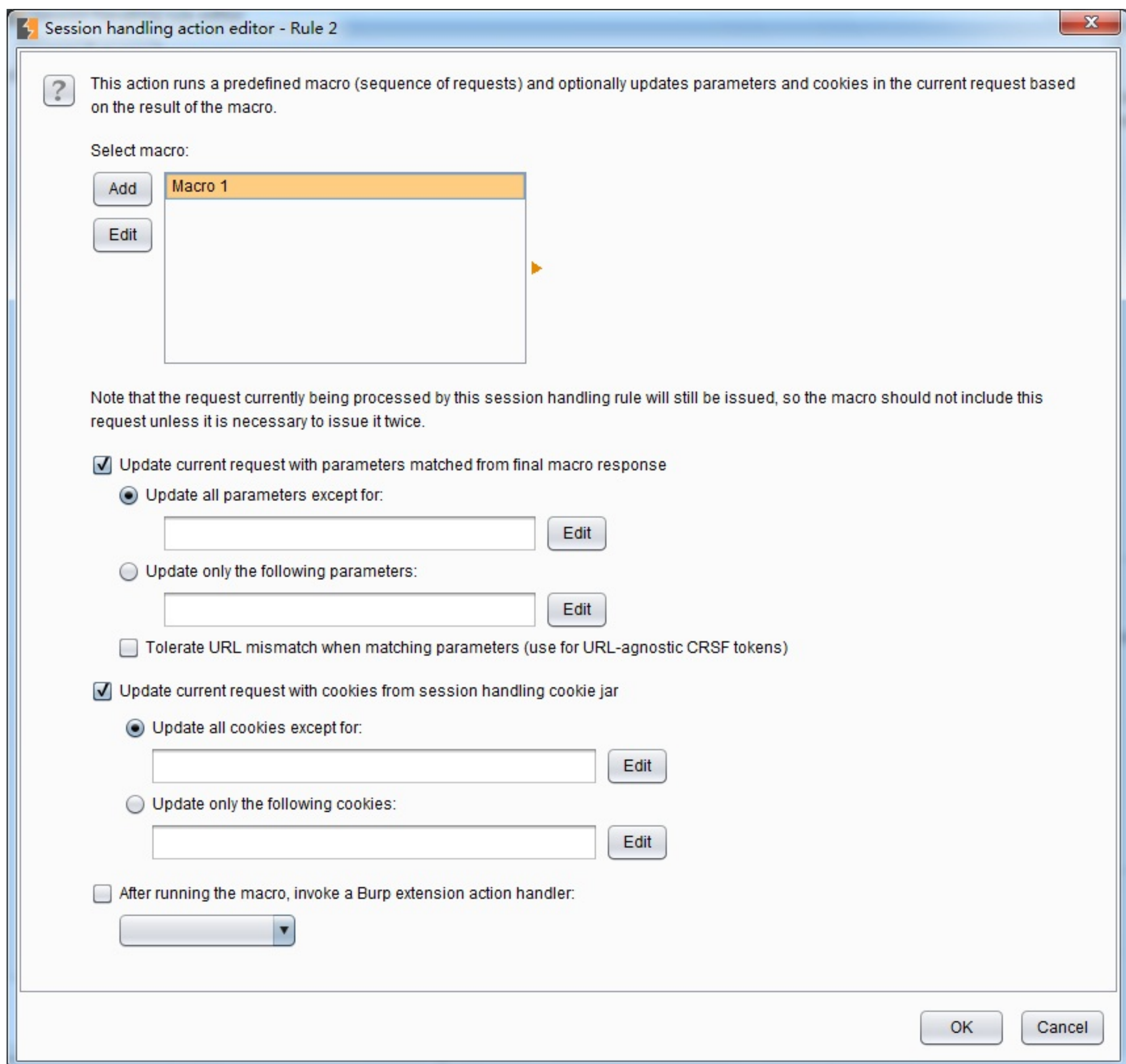


当Burp发出请求，并验证了会话的有效性之后，将不再做下一步动作；如果运行了宏，则Burp将进一步检查请求的应答消息。为了准确地确定会话有效性，我们通常将Burp检查响应配置为搜索表达式，其搜索范围为（==上图中3部分所示==）：a)HTTP响应头 b)HTTP响应体 c)任何重定向目标的URL 除了范围外，在设置正则匹配/字符匹配的字符串同时，我们也可以匹配大小写是否敏感、会话是否有效、如果会话失效，需要做的下一步动作是什么等操作。 关于会话失效后的下一步操作，Burp中预制了两个类型，如下图所示：

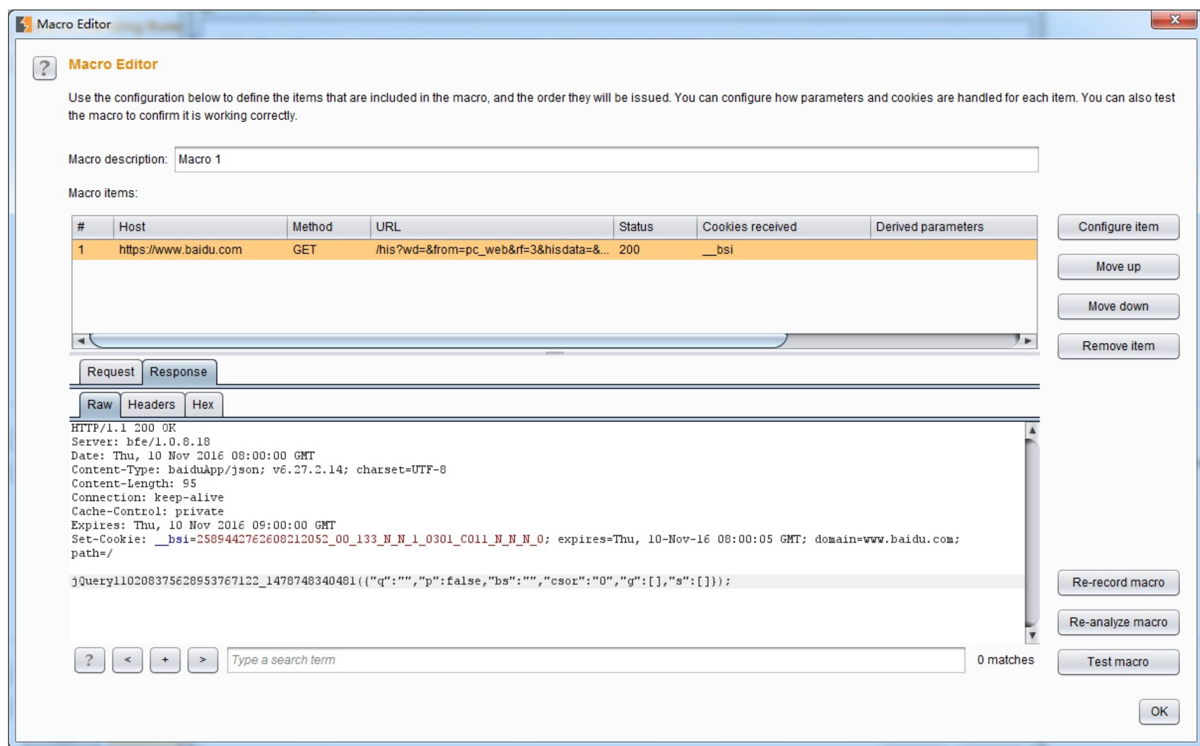


a) 运行宏 b)从浏览器内部恢复会话 针对于这两类操作，会在接下来的章节中描述，此处不再赘述。

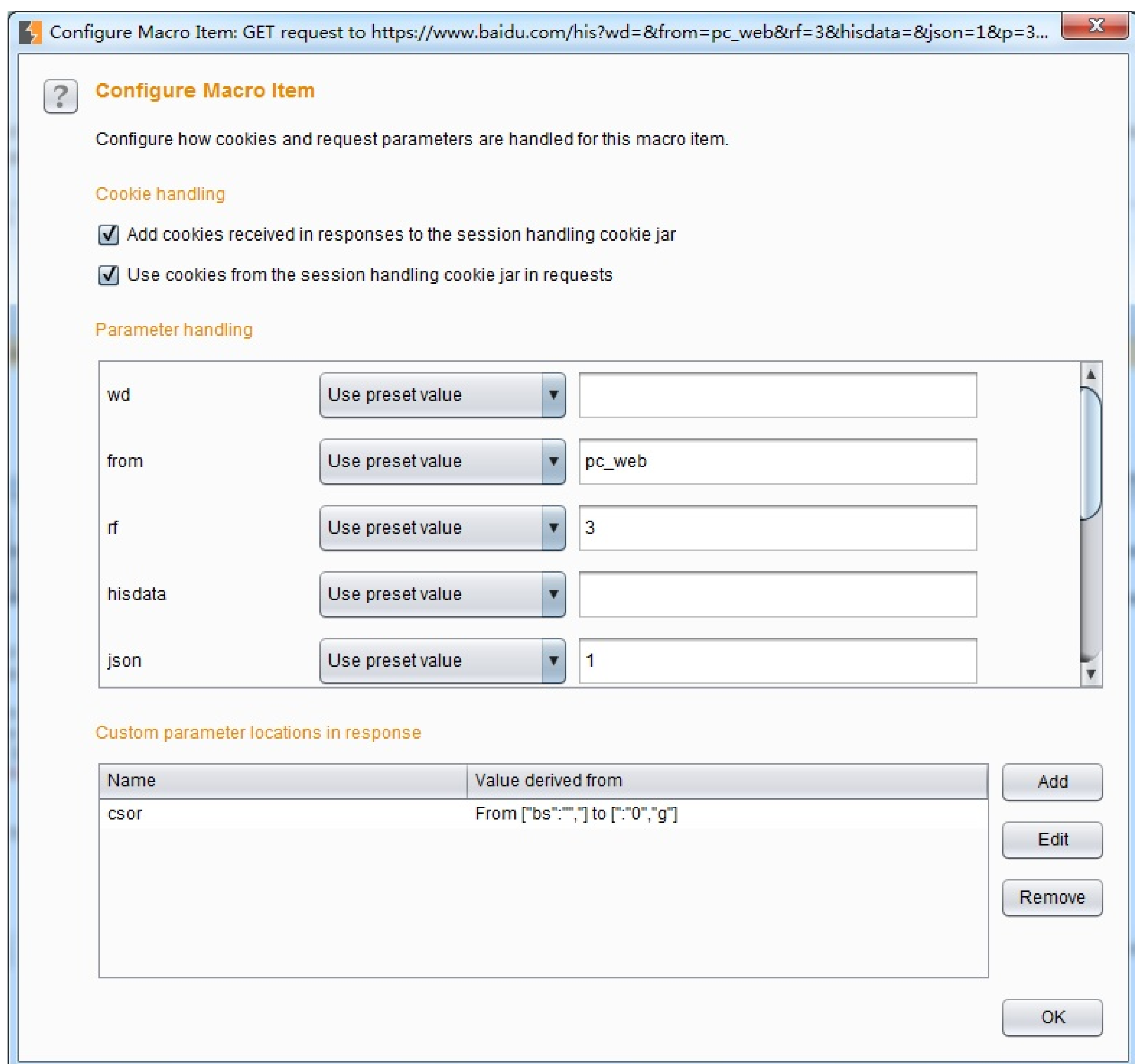
4. **Prompt For In-Browser Session Recovery** 这个配置的动作是针对于会话失效后，从浏览器内部进行会话恢复的。在会话恢复时，需要使用Proxy代理的请求记录信息，如果使用此动作，则浏览器的代理设置与Burp需要一致。
5. **Run a Macro** 在Burp中，宏是一系列顺序操作的Burp操作的总和，预先定义好的，在Session中被运行，用于会话规则的处理。宏运行后，Burp根据最终的宏响应报文来选择更新当前正在处理的请求中的参数和Cookie。至于宏的定义和设置在接下来的章节中会专门描述,此处仅做简要介绍。 当我们在添加Rules Action时选择了“Run a Macro”项，则弹出的宏配置界面如下图所示：



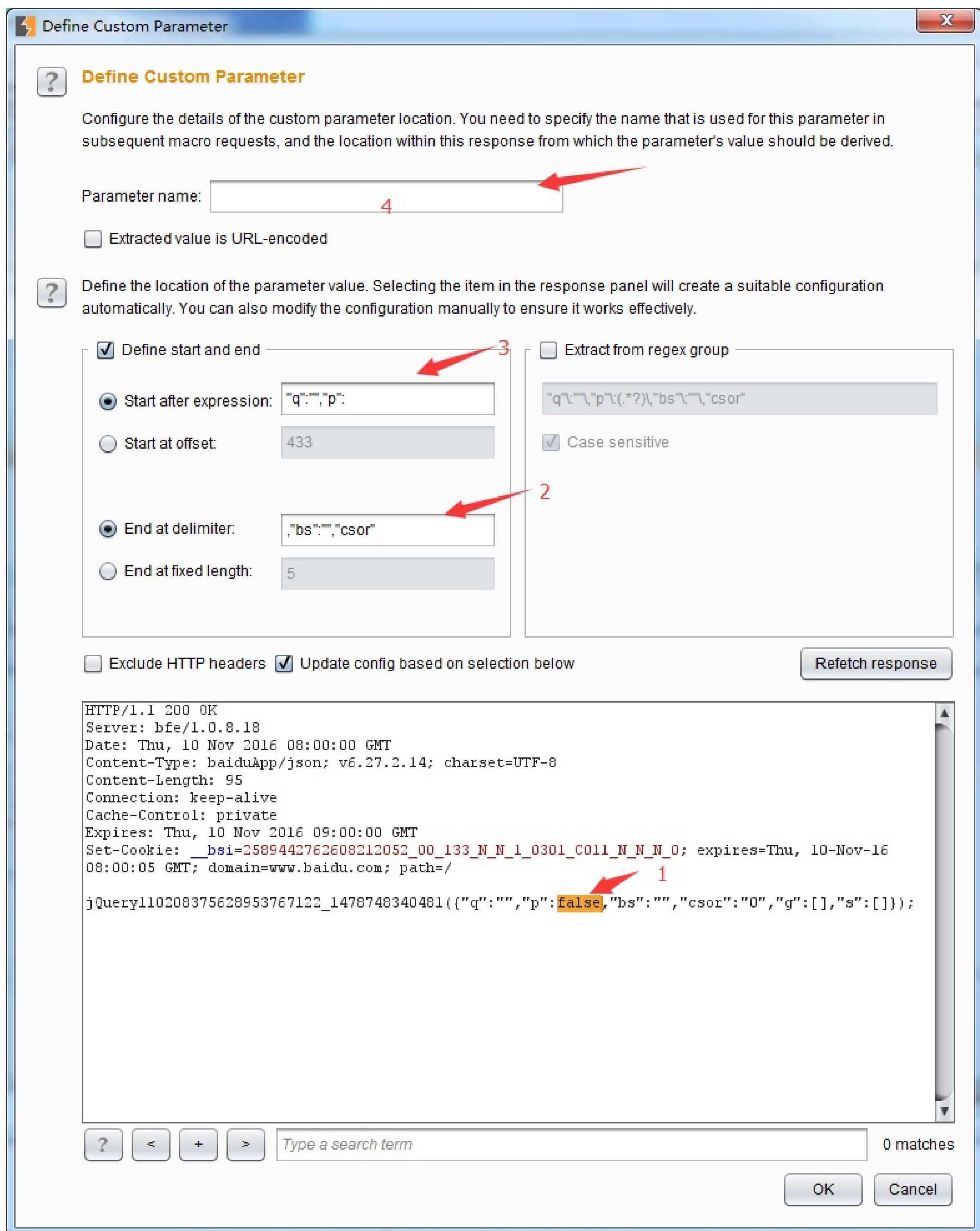
点击【Add】则添加一个宏，选择某个宏记录，点击【Edit】则可以对宏配置进行编辑。其设置界面如下图：



上图中宏的名称、items、请求和应答消息等简单关注即可，需要重点关注的是【configure item】按钮中对参数的设置。当我们点击此按钮，打开宏参数的配置界面：



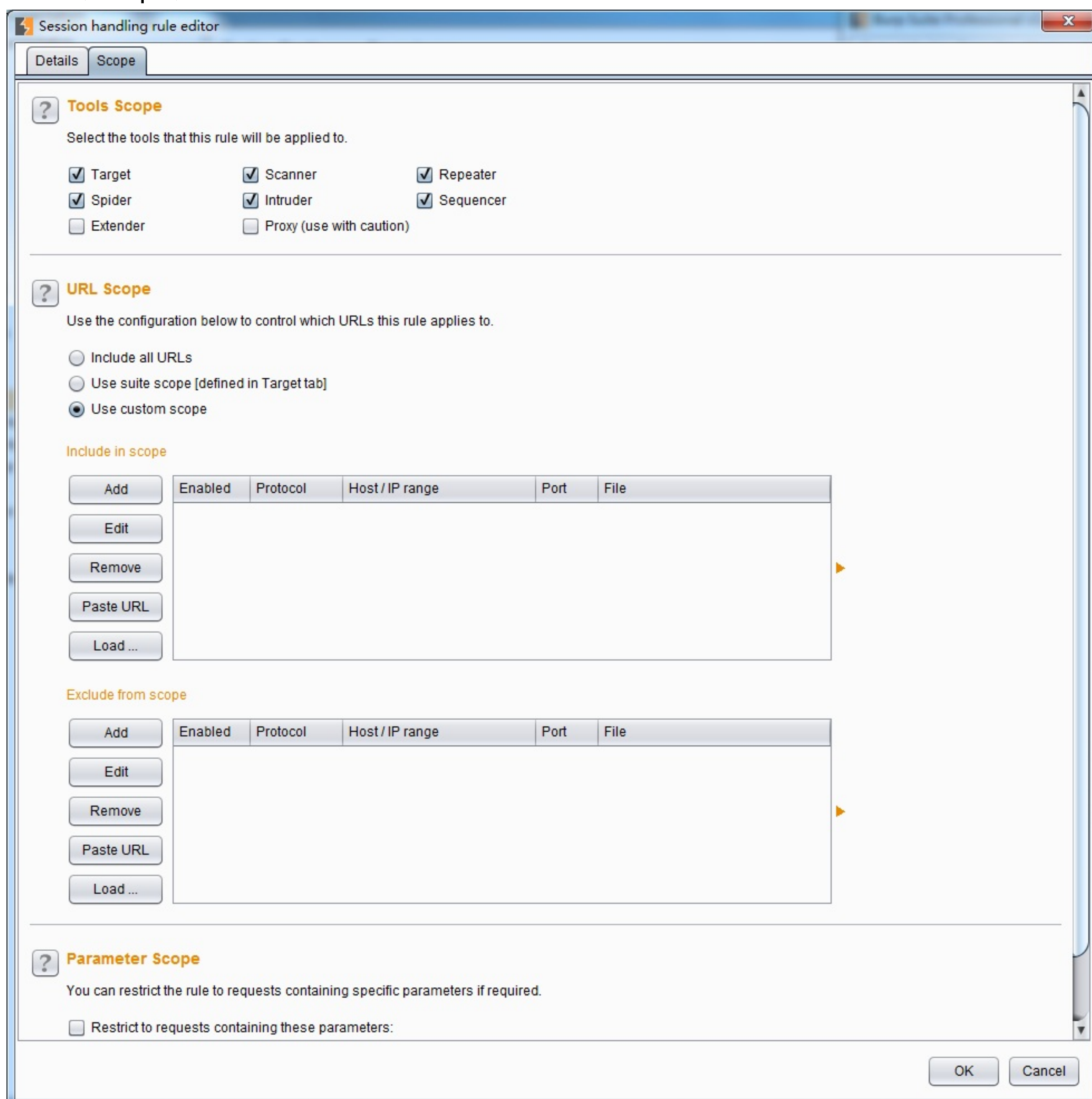
此界面上已经对请求报文中的参数和cookie自动提取出来，按照元素分别展示，同时，界面下半部分为客户化参数设置，可以自定义自己想要的参数，并从应答报文中提取参数的值。



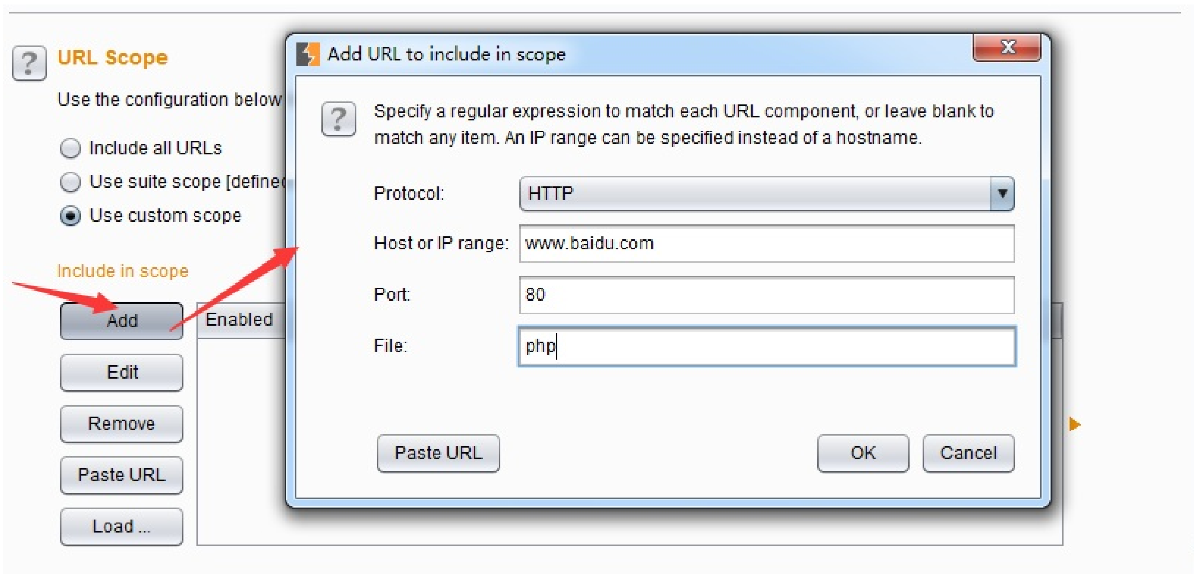
在上图中，当我们鼠标双击1处时，2和3处会自动设置提取数据的段，我们只要在4处简单填写参数的名字即可完成常用的宏参数设置。 设置完宏之后，当宏运行时，其作用的范围依赖于Session Scope 的设置。

6. **Run a Post-Request Macro** Post-Request宏通常使用于多步骤测试的场景，例如：后一步的测试数据依赖于上一步的请求结果。在这些场景下，Post-Request宏的使用会帮助你完成参数值的自动化地填充、fuzz、scan等。
7. **Invoke a Burp Extension** 这个配置的动作是Burp的拓展插件，来对当前会话数据进行处理。此处调用的插件，必须要先在Burp的插件中心进行注册。关于Burp插件，请阅读《Burp Suite应用商店插件的使用》章节。

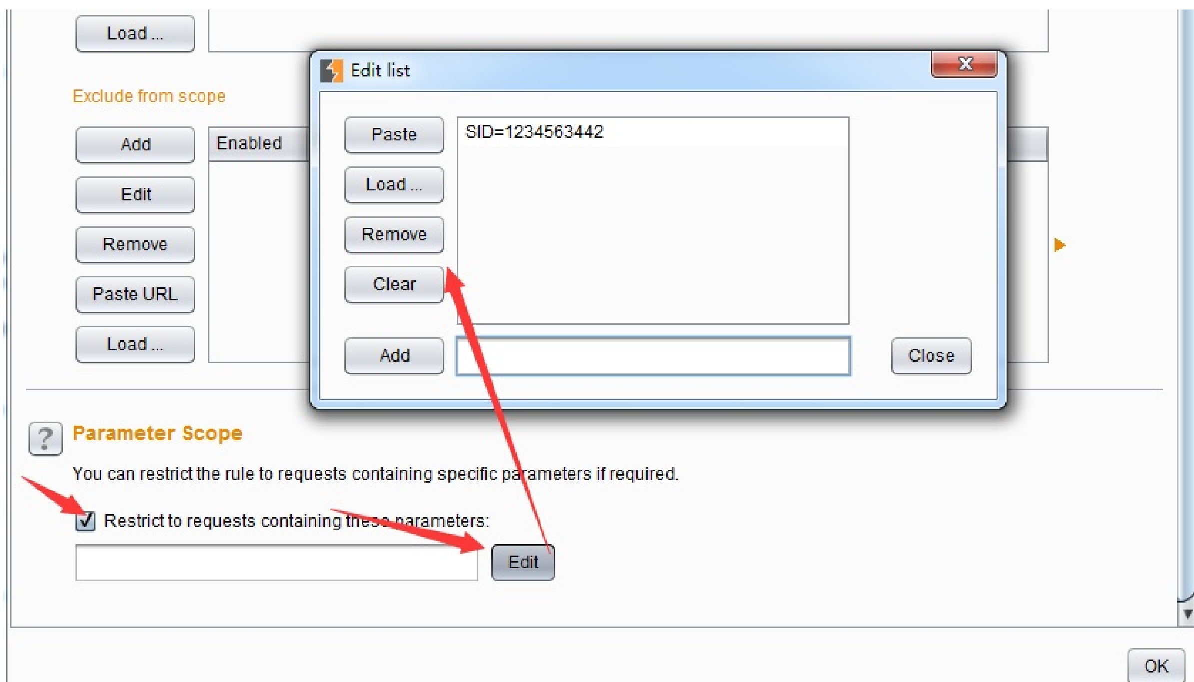
8. 范围（Scope）



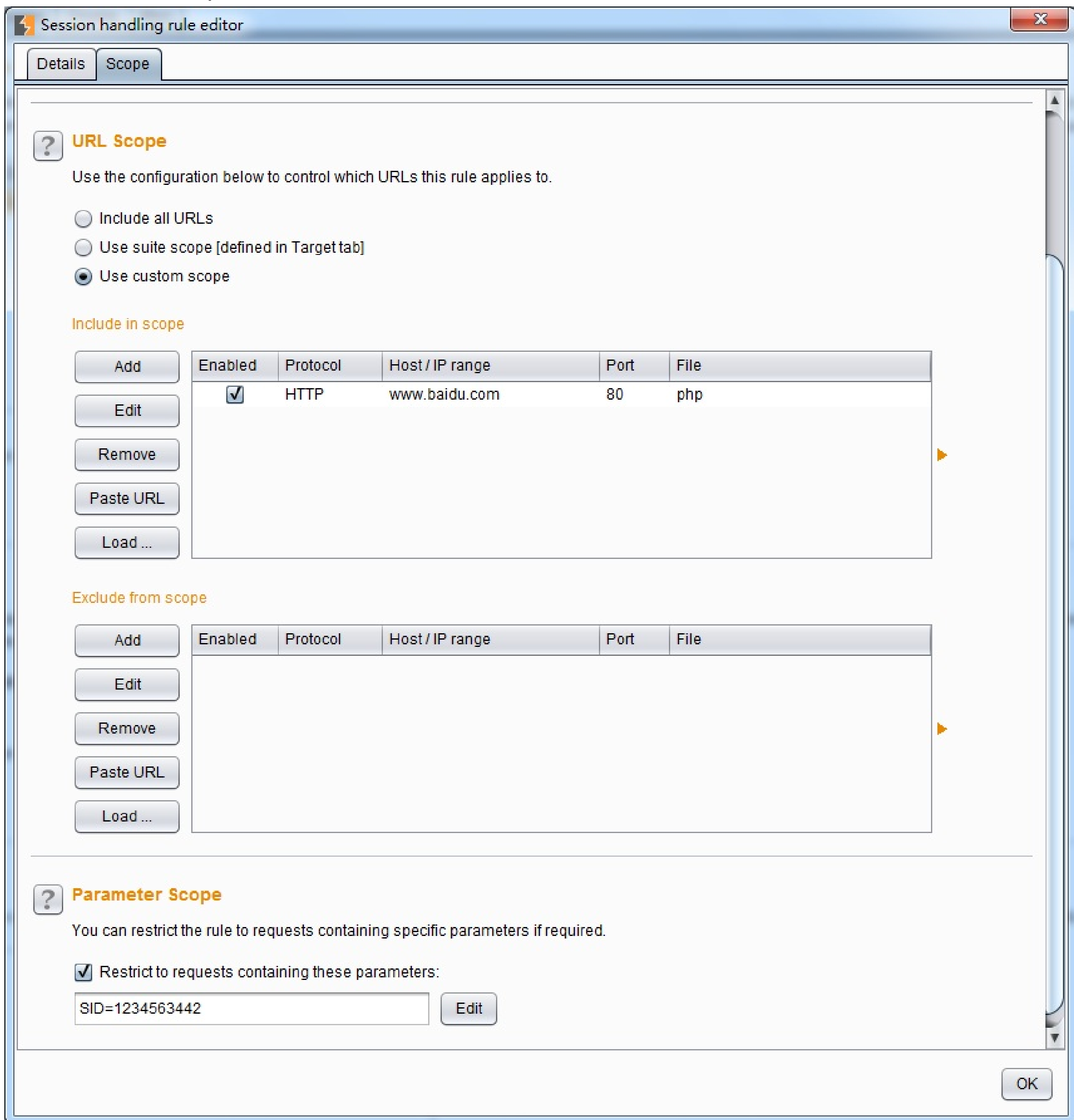
而对于Burp做出的每个请求，它在Scope中定义规则在哪些请求的范围内，并且按顺序执行所有这些规则的动作（除非条件检查动作确定不应该对请求）。 每个规则的范围可以基于正在处理的请求的以下特征来定义，在Scope面板中共分为以下三类： 1.正在发送请求的**Burp**工具（**Tools Scope**），包含Burp的各个常用工具组件，例如：Target、Scanner、Proxy、Intruder等。 2.请求的网址（**Urls Scope**），包含所有的URL地址、指定的作用于、自定义作用域三种方式，其配置与Target类似。



3.请求中的参数名称（**Paramters Scope**），当选中此项时，点击【Edit】按钮即可对参数进行配置，如下图所示例：



配置完毕后的Scope截图大体如下图所示：



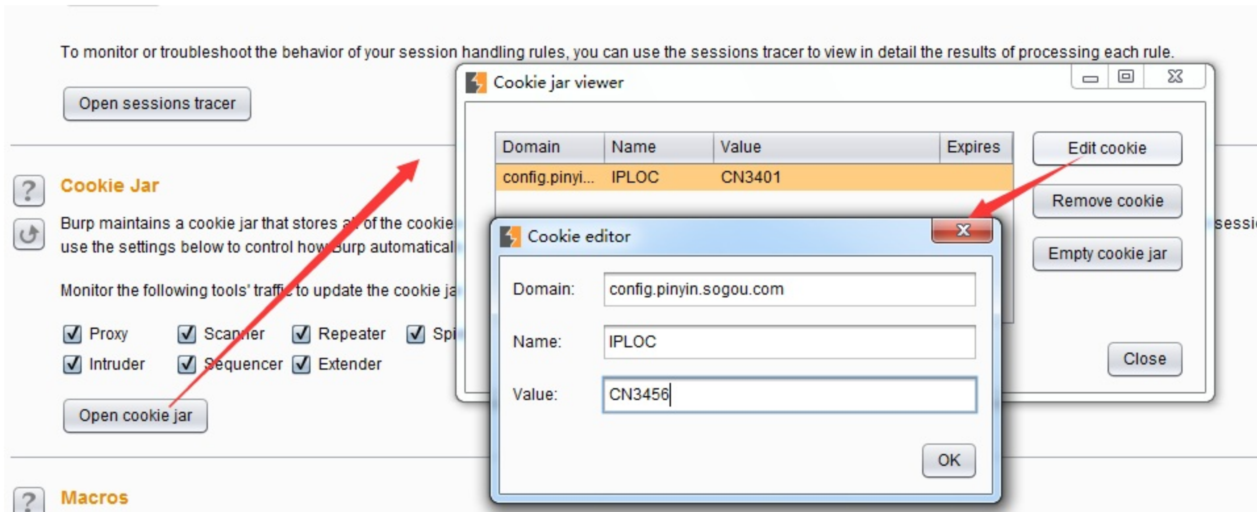
配置完成后，会话处理规则将对作用域的Burp工具组件中的会话进行处理，例如，如何配置了Proxy，则通过Proxy的会话，可以通过此面板下方的【open sessions tracer】进行会话跟踪。如下图：

Cookie Jar

Burp通过维护Cookie jar来维护你访问过得所有web站点的cookie信息，Cookie jar的信息在Burp的所有工具组件之间是数据共享的。

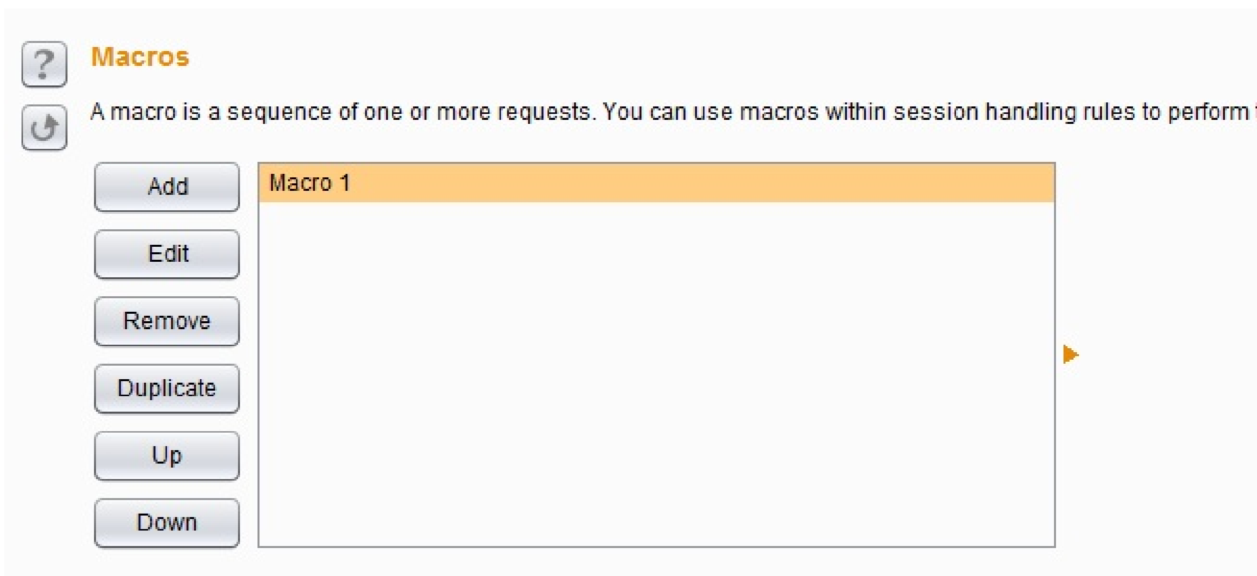


我们可以通过上图中的勾选项配置，来指定Cookie jar在哪些工具组件之间生效。当设置完毕后，这些工具组件的流量数据更新，会保证Cookie jar的数据也一致性的更新。同时，我们也可以点击下方的【Open cookie jar】按钮，来做cookie信息的手工维护。



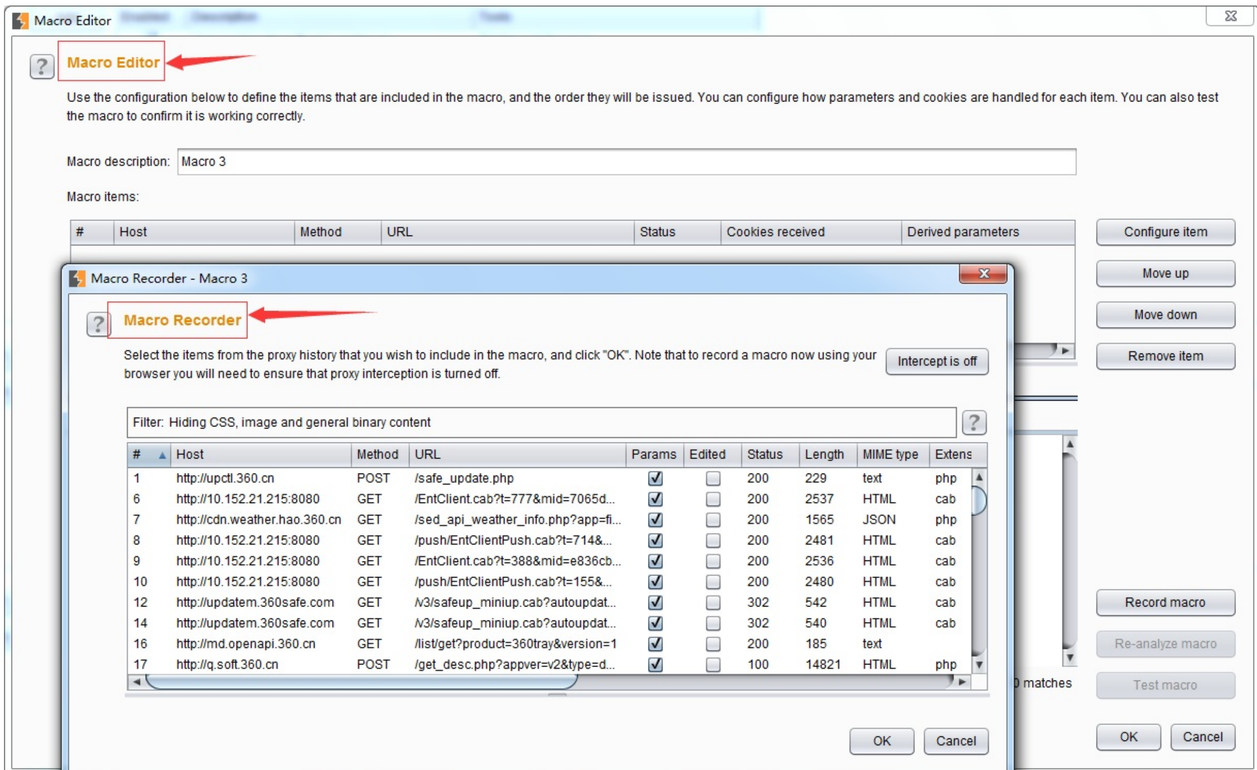
宏(Macros)

在会话处理规则章节中，我们队宏的定义已经做了初步的描述，现在我们就来讲一讲Burp的宏的使用。 Burp中宏的定义是：一个或者多个请求的预定义序列，其本质是一个或者多个请求，按照一定的顺序组成并按照顺序执行的操作集合的总称。典型的宏的使用场景有： **a)**检测用户登录页面，判断当前会话是否仍然有效。 **b)**模拟登录操作，以获取一个新的会话令牌。 **c)**在多步骤测试过程中，获取前一步骤的反馈数据，作为后面测试的输入数据。 **d)**在多步骤测试过程中，完成测试目的后，用于结果的验证。 除了基本的请求序列外，宏还包含每一个请求相关的cookie、请求参数、数据依赖等配置项。 **1.** 宏的维护

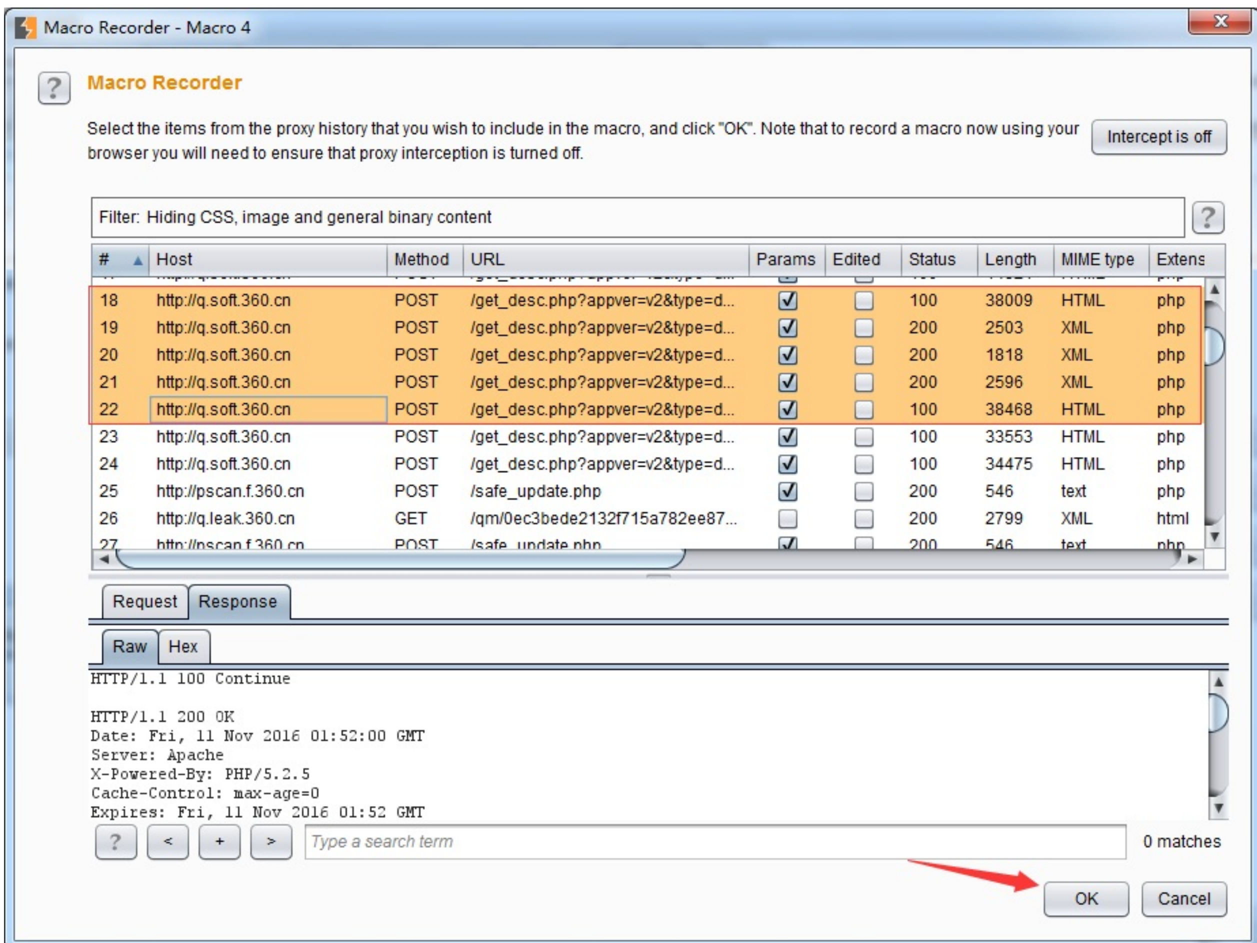


上图为宏的维护界面，通过【Add】、【Edit】、【Remove】按钮，我们可以对宏进行新建、修改和删除操作。当有多个宏的时候，我们可以通过【Up】和【Down】按钮来调节宏的位置，来控制宏执行的先后顺序。 **2.** 宏的新建和修改 新建是新增一个宏，修改是对宏列表中已有宏的信息进行修改，其界面和操作类似。此处仅以新建为例，来讲述宏的使用。当我们

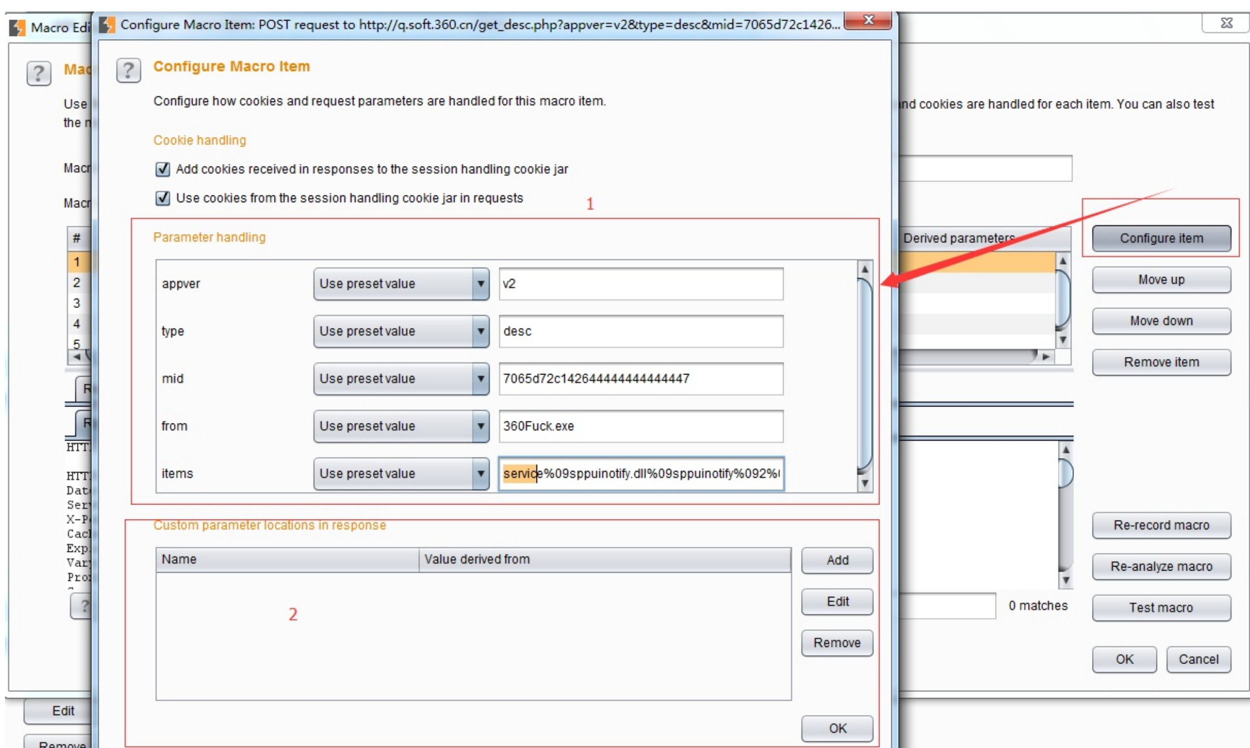
点击【Add】按钮来新建一个宏，则Burp将弹出宏信息录入界面。



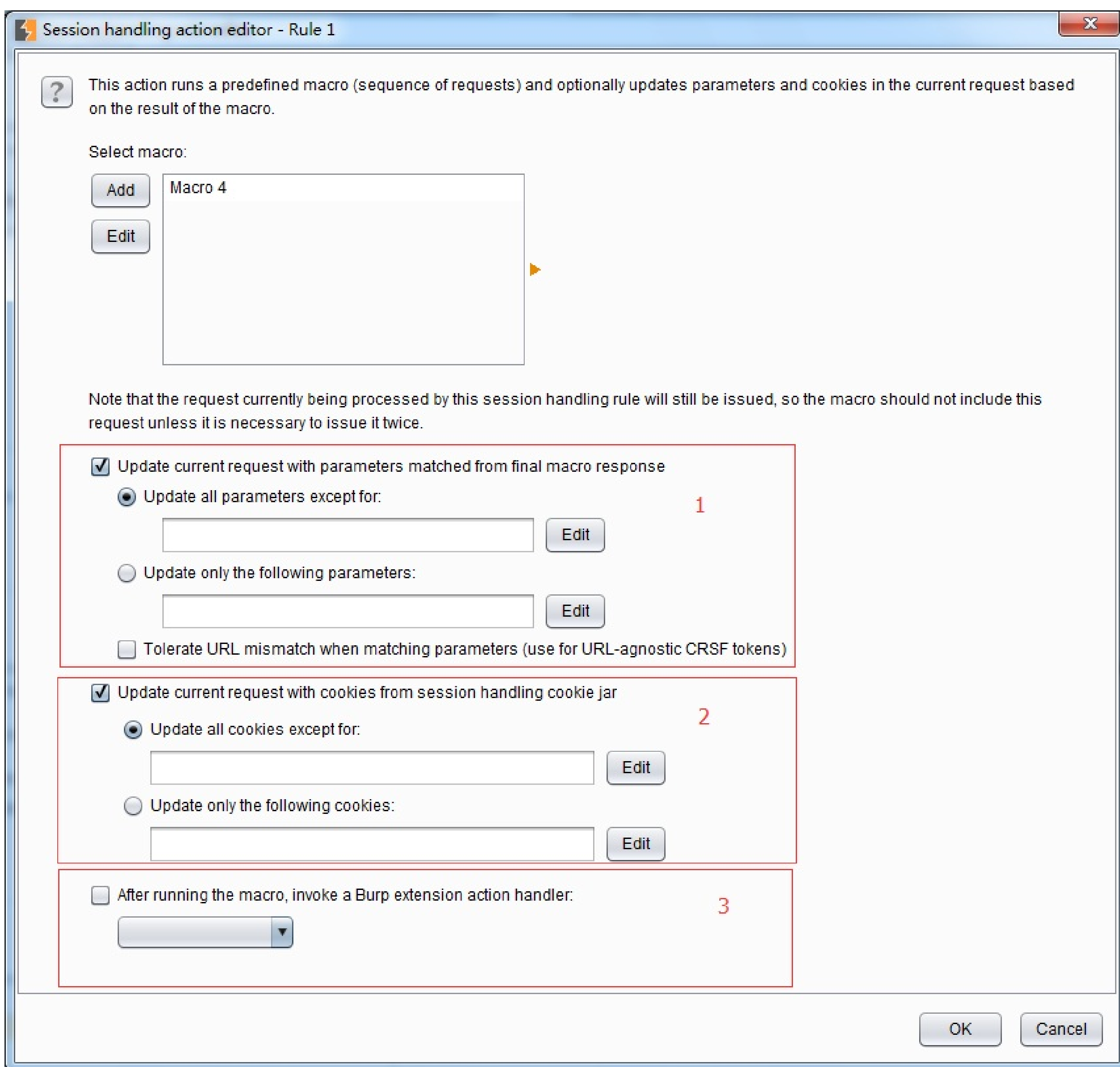
宏信息的录入界面为图中的Macros Editor，而图中的Macros Recorder界面为请求的记录。当我们新建宏操作时，可以选择一条或一组的请求记录，做为宏的基础。



如上图所示，选择序列18~22的记录作为宏的序列，点击【OK】保存序列后，配置参数信息。



当我们点击【configure Item】按钮时，即弹出参数配置界面（如上图）。其配置界面分上下两个部分，上部为图中1所示，主要是对已有参数值的设置，下部为图中2所示，我们可以根据实际场景的需要，添加自定义参数和参数值。 完成了如上的设置之后，我们点击【OK】按钮，则一个宏已经被正确的创建。 **3.** 宏的使用 完成宏的设置之后，下面我们就看看宏在渗透测试中通常是被如何使用的。在会话处理规则（Session Handling Rules）章节中我们知道，配置【Rule Actions】时有**Run a Macro**、**Run a Post-Request Macro**两个选项，当我们设置了其中的选项，针对于当前会话，在作用域的范围内，宏就会生效。无论你设置了哪种类型的宏，其使用的数据处理逻辑大体如下图所示：

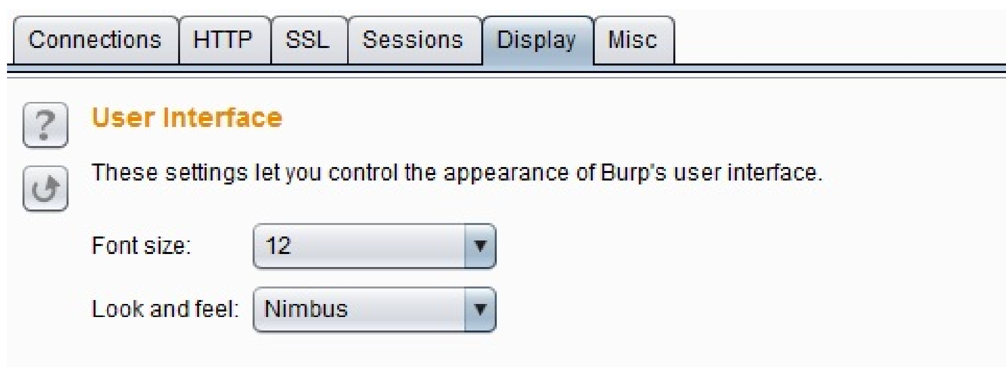


其中图中1所示为通过宏应答的响应更新参数的值，我们可以全量更新参数值也可以部分更新参数值；图中2所示为更新cookie的值，同样，我们也可以全量更新参数值也可以部分更新参数值；图中3所示为执行宏之后，还可以执行Burp的插件，需要执行的插件即在此处配置。

显示设置（Display）

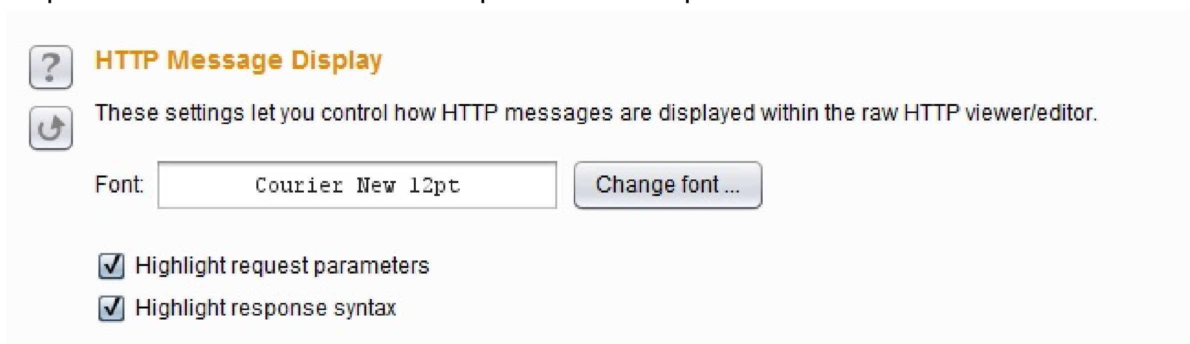
和其他的软件一样，Burp也存在显示设置，作为软件与用户习惯交互的接口。 Burp的显示设置主要包含：用户界面（User Interface）、Http消息显示（HTTP Message Display）、字符集设置（Character Sets）以及页面渲染（HTML Rendering）

- 用户界面主要用来设置字体和界面风格

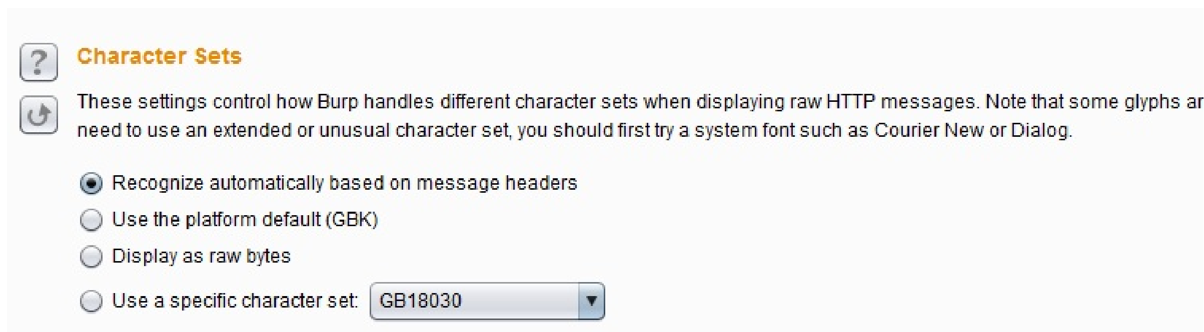


常用的有Windows风格、Windows经典风格、Nimbus等，修改配置后，需要重启Burp才会生效。

- Http消息显示主要用来设置其他Burp工具组件中http消息的显示字体、高亮等形式。



- 字符集设置主要用来设置http消息显示时使用的字符集编码，正确的使用字符集是防止消息显示乱码的基础，默认情况下会自动获取系统字符集。

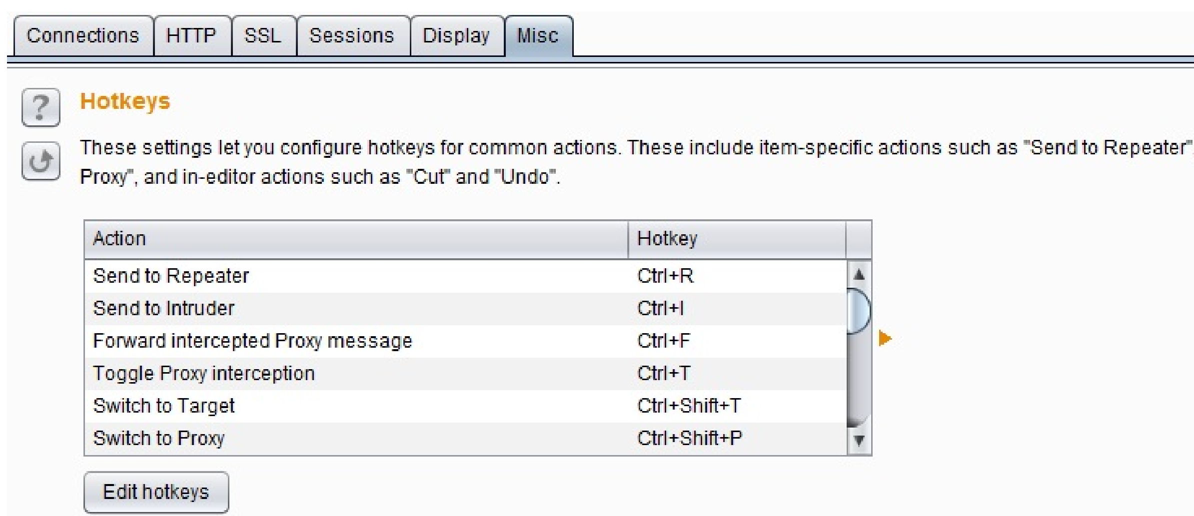


- 页面渲染是指http消息进行渲染时，是否也显示图片等信息，如果显示图片，可能会增加新的http请求消息。

杂项设置（Misc）

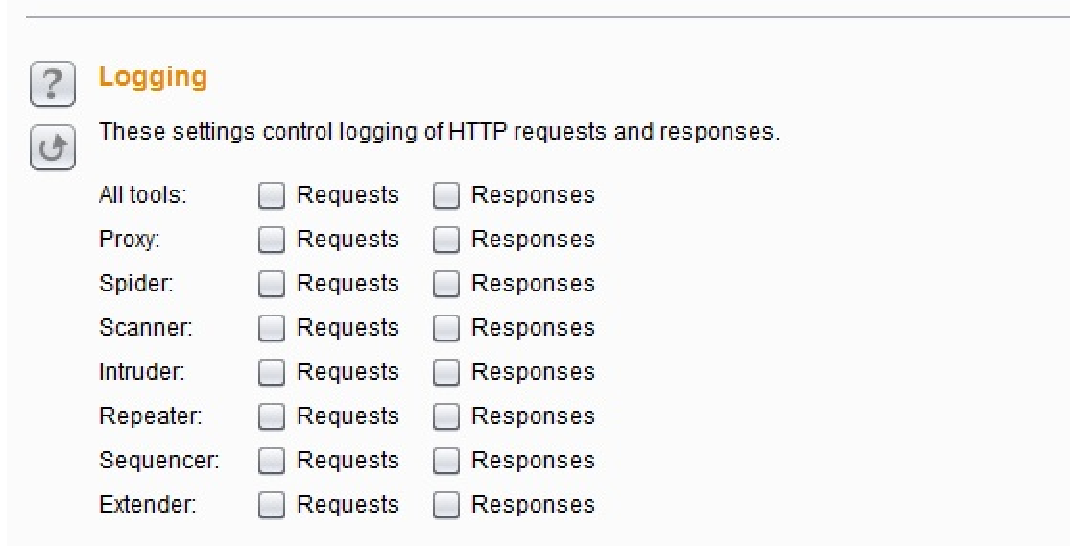
Burp的杂项主要包含以下七个部分内容：

- 快捷键设置（Hotkeys）



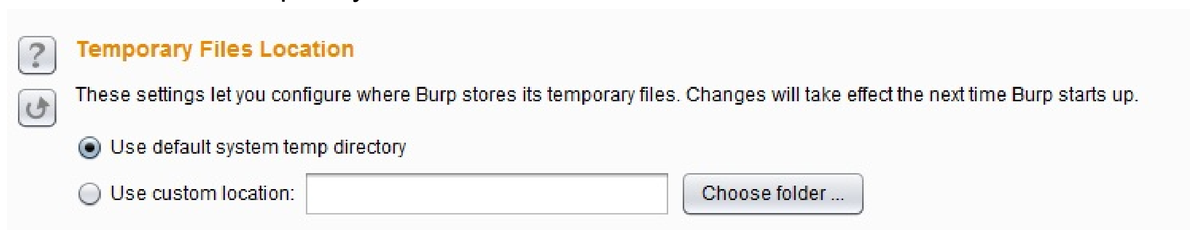
Burp的快捷键设置遵循了系统软件的设置习惯，比如Ctrl+V、Ctrl+C、Ctrl+Z都是和操作系统一样，同时，在各个工具组件之间的切换和消息传递时，Burp的快捷键基本遵循了Ctrl+组件的首字母，例如： send to Repeater是Ctrl+R send to Intruder是Ctrl+I 详细的快捷键读者自己在使用过程中，会慢慢熟悉，而且，Burp也提供了自定义快捷键的功能，只有点击下方的【Edit hotkeys】按钮，进行修改即可。

- 日志设置（Logging）



用来控制Burp中的哪些工具组件需要记录日志，记录时，也可以单独记录请求或者应答消息。

- 临时文件位置（Temporary Files Location）



默认情况下，burp会在用户的系统目录作为临时文件的目录，同样，我们也可以修改这个目录，指定其他的盘符目录作为临时文件目录，burp在工作过程中，产生的临时数据会存放在此目录中。如果修改了此设置，需重启Burp后方可生效。

- 自动备份设置（Automatic Backup）

Automatic Backup

The automatic backup feature can be used to save a copy of Burp's state periodically in the background, and on exit.

Automatically backup state every 30 minutes

To folder: Choose folder ...

Include in-scope items only

Backup on exit

此设置用于保存Burp的状态和配置，设置完成后，会在后台定时地保存Burp的当前配置参数和运行状态。

- 任务队列（Scheduled Tasks）

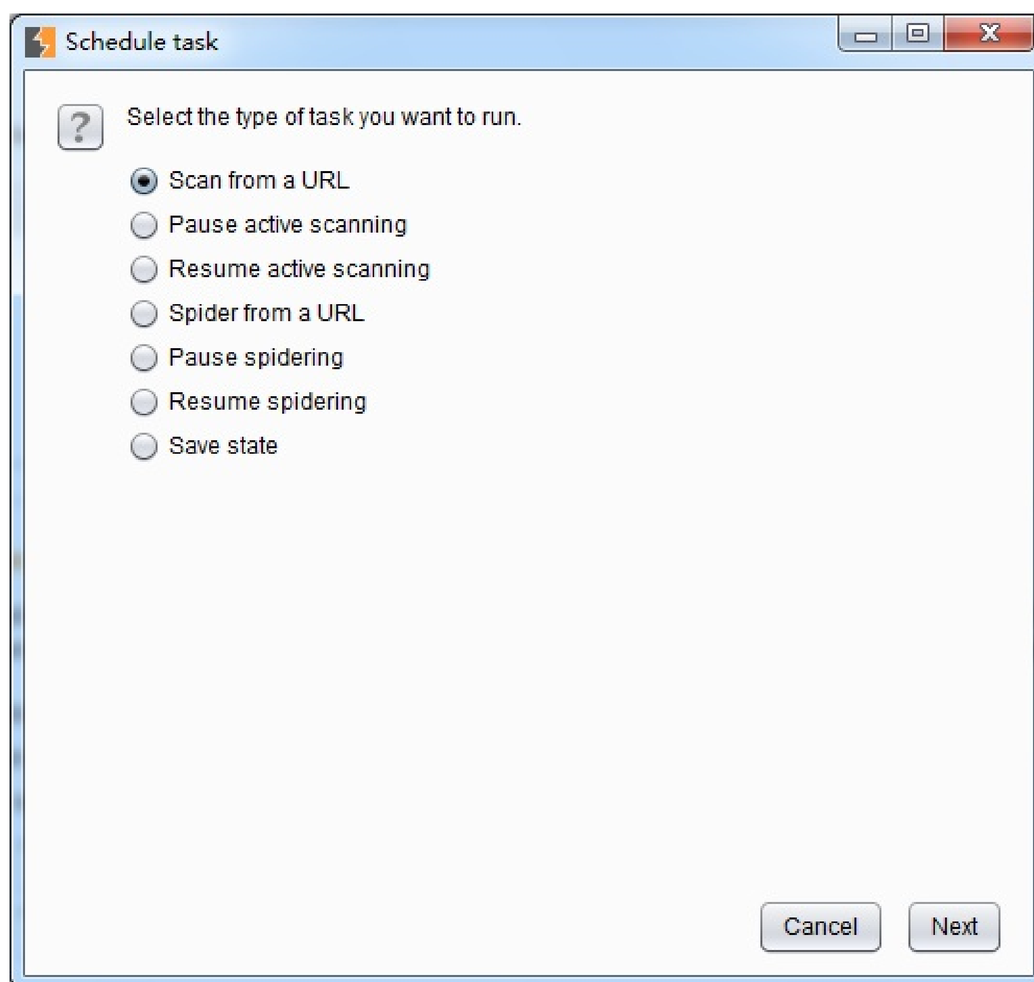
Scheduled Tasks

These settings let you specify tasks that Burp will perform automatically at defined times or intervals.

Add Edit Remove

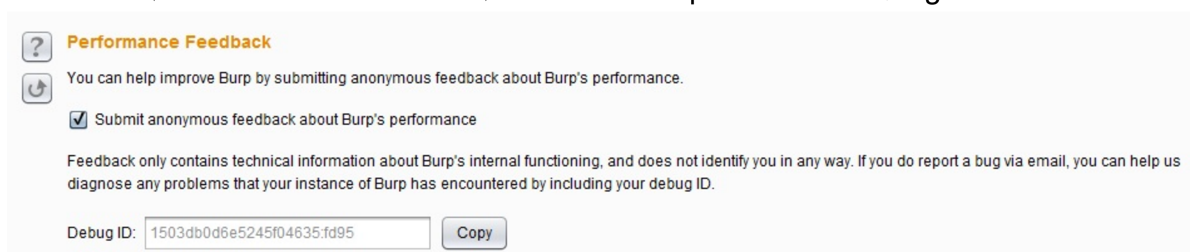
Time	Repeat	Task

我们可以通过任务队列的管理，来控制任务的开始和结束以及周期性运行。目前Burp的任务控制主要为以下几类（如下图），点击【Add】按钮，按照操作向导一步步的执行即



可。

- 性能反馈（Performance Feedback） 主要用于Burp的使用问题或bug反馈。



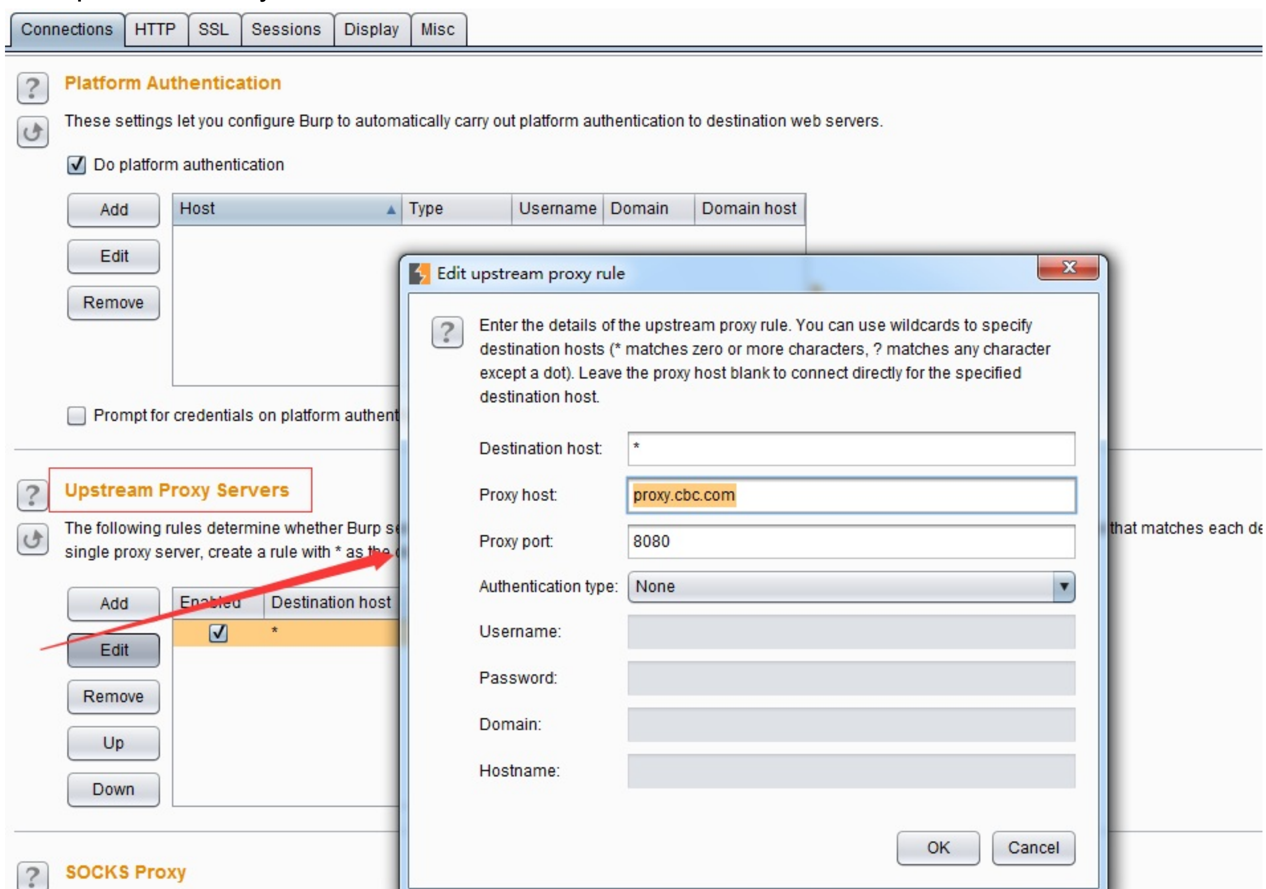
Burp Suite应用商店插件的使用

Burp在软件中提供了支持第三方拓展插件的功能，方便使用者编写自己的自定义插件或从插件商店中安装拓展插件。Burp扩展程序可以以多种方式支持自定义Burp的行为，例如：修改HTTP请求和响应，自定义UI，添加自定义扫描程序检查以及访问关键运行时信息，包括代理历史记录，目标站点地图和扫描程序问题等。 本章讲述的主要内容有：

- 应用商店插件的安装使用（BApp Store）
- 管理和加载Burp 插件（Extension）
- 其他选项设置（Options）

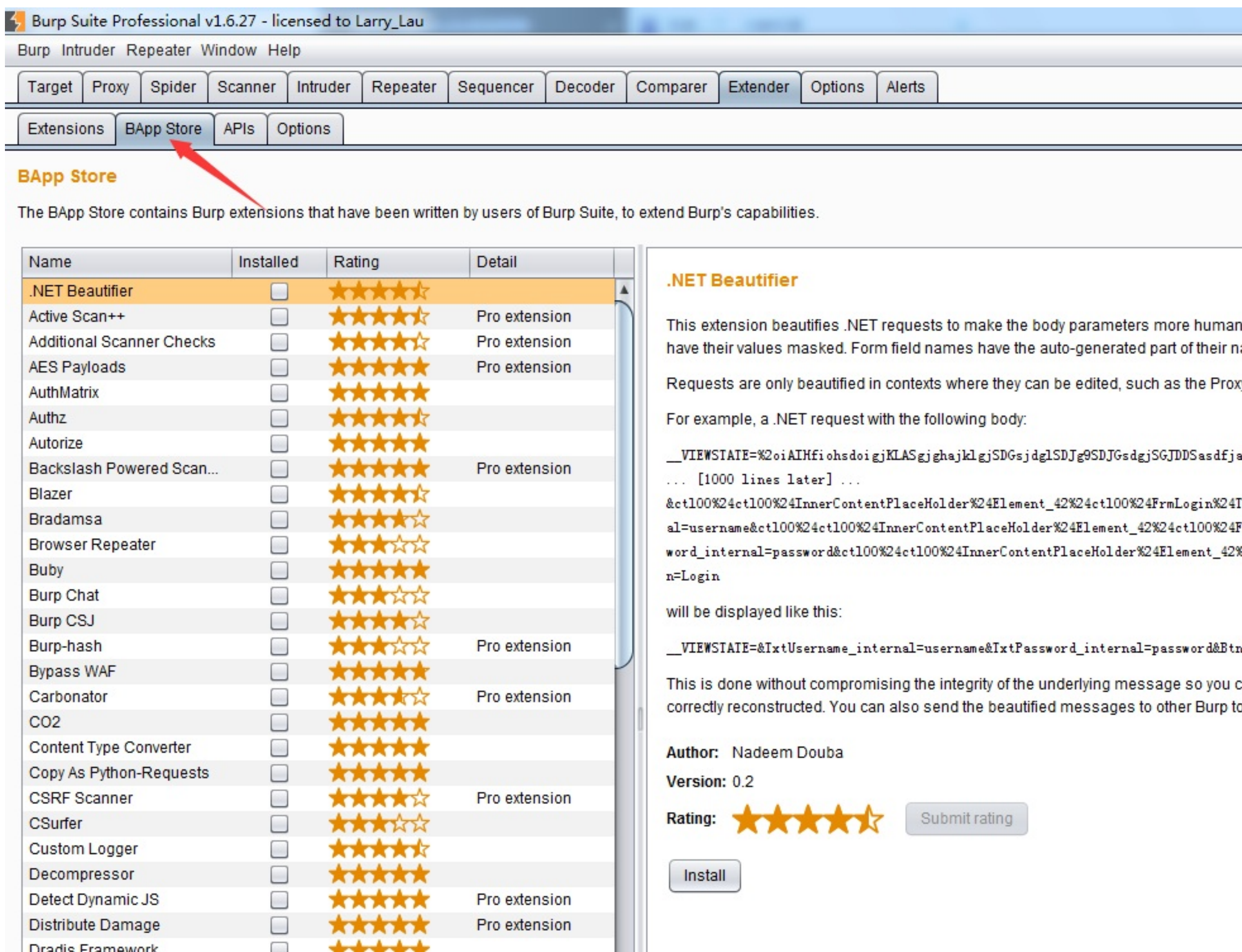
应用商店插件的安装使用

在Burp Extender 面板中，有一个BApp Store的Tab页，这就是Burp的应用商店，内容是提供各种Burp的插件。 默认情况下，当你点击【BApp Store】的Tab页时，界面列表会显示插件明细，若你的环境是通过代理访问外网的，则需要在【Options】->【Connections】->【Upstream Proxy Servers】进行设置，具体如下图所示：

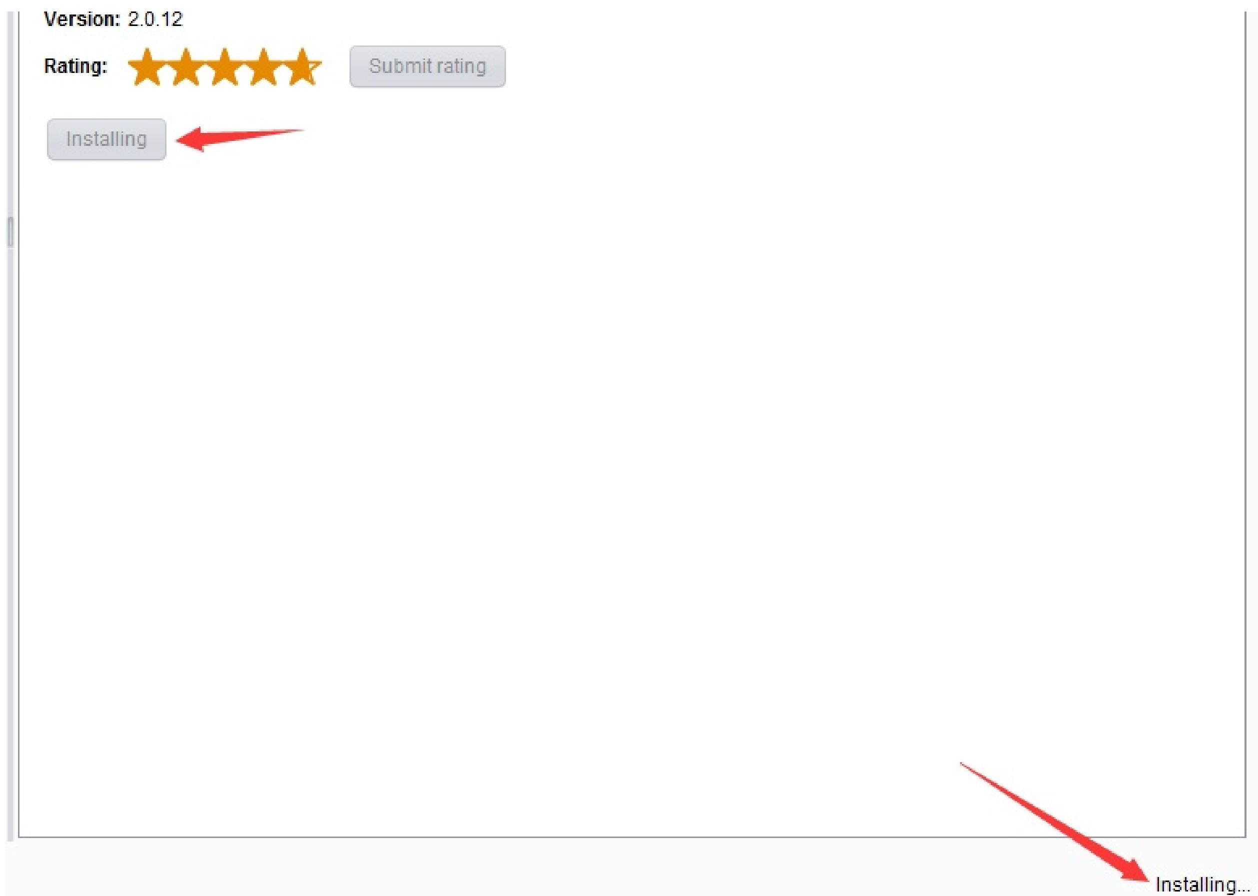


其中代理服务器的host和port为你本地的网络环境访问外网的代理主机和端口，更详细的设置请参加Connections章节相关内容。

如果你的网络设置没有问题，则应用商店的界面显示大体如下：



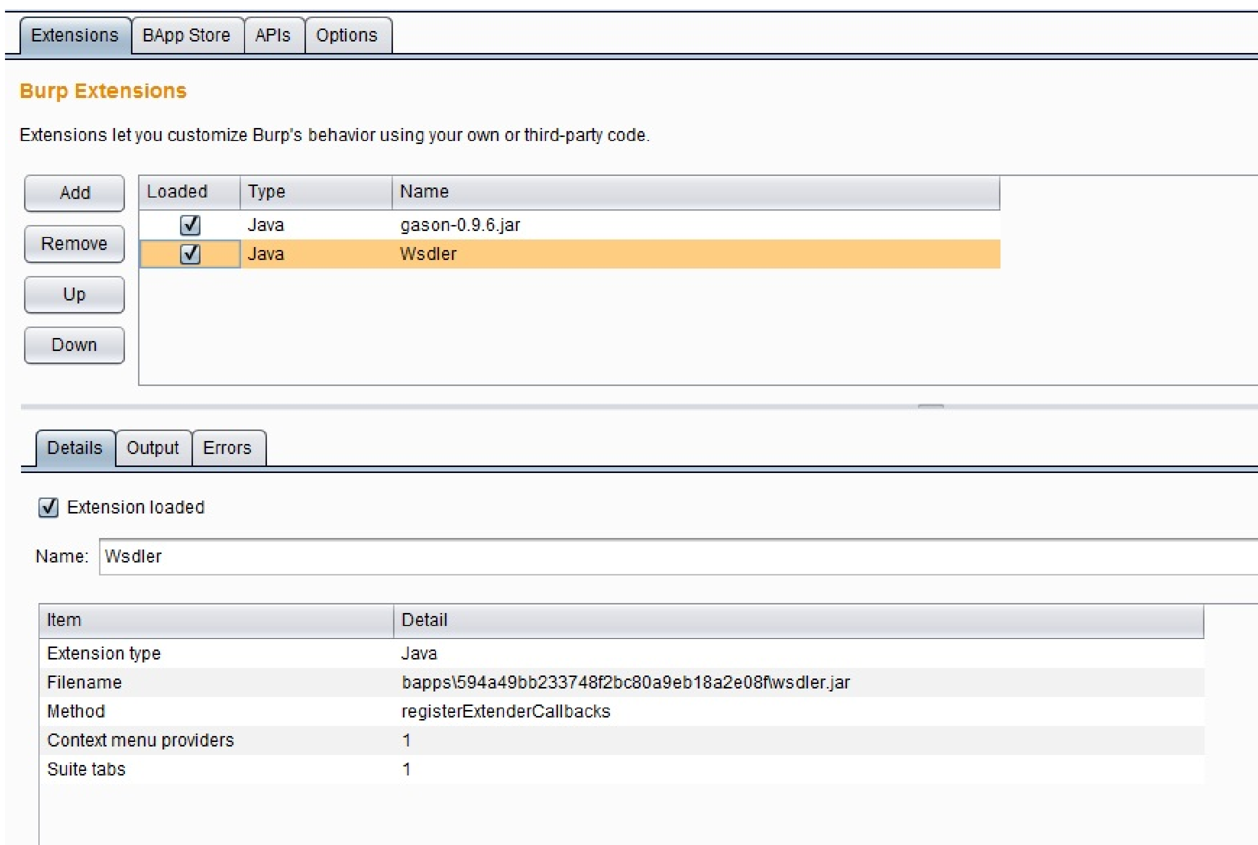
从图中我们可以看出，左边为各个插件的应用列表，当选中某个插件后，右侧显示的为该插件的描述信息和安装信息。如果我们需要使用某个插件，则点击右侧下方的【install】按钮，进行安装。



此时，安装按钮置为灰色，同时显示为【installing】，右下角也显示安装中，如上图。安装完成后，界面会显示重新安装【Reinstall】和插件评分按钮【Submit rating】，作为插件商店的用户推荐。



安装完毕后，在Burp Extender 的Extension的Tab页面中，会自动显示已加载的插件列表。通过插件列表的管理，我们可以对插件进行后期的维护。



当然，除了从应用商店自动安装插件外，我们也可以下载插件，进行手工安装。如下图：

BApp Store

The BApp Store contains Burp extensions that have been written by users of Burp Suite, to extend Burp's capabilities.

Name	Installed	Rating	Detail
Logger++			
Manual Scan Issues			Pro extension
MindMap Exporter			
NMAP Parser			
Notes			
Paramalyzer			
ParrotNG			Pro extension
Payload Parser			
Pcap Importer			Pro extension
PDF Metadata			Pro extension
PDF Viewer			
Protobuf Decoder			
Python Scripter			
Random IP Address Header			
Reflected Parameters			Pro extension
Reissue Request Scripter			
Report To Elastic Search			Pro extension
Request Randomizer			
Retire.js			Pro extension
SAML Editor			
SAML Encoder / Decoder			
SAML Raider			
Sentinel			
Session Auth			Pro extension
Session Timeout Test			
Site Map Fetcher			
Software Version Reporter			Pro extension
SQLiPy			Pro extension
ThreadFix			Pro extension
WCF Deserializer			
WebInspect Connector			Pro extension
WebSphere Portlet State D...			
What-The-WAF			
WSDL Wizard			
Wsdler 1	✓		
XSS Validator			

Refresh list　Manual install ...

Wsdler

This extension takes a WSDL request, parses out the operations that are associated with the targ SOAP requests that can then be sent to the SOAP endpoints.

Select BApp File

查看: Burpsuite_pro

EnCode.exe　jython-installer-2.7.0.jar
gason-0.9.6.jar　jython-standalone-2.7.0.jar
GrabTencentExmailContacts.jar　run.bat
JavaScriptInjector.py
v1.6beta.jar　javasnoop-master.zip

文件名: gason-0.9.6.jar

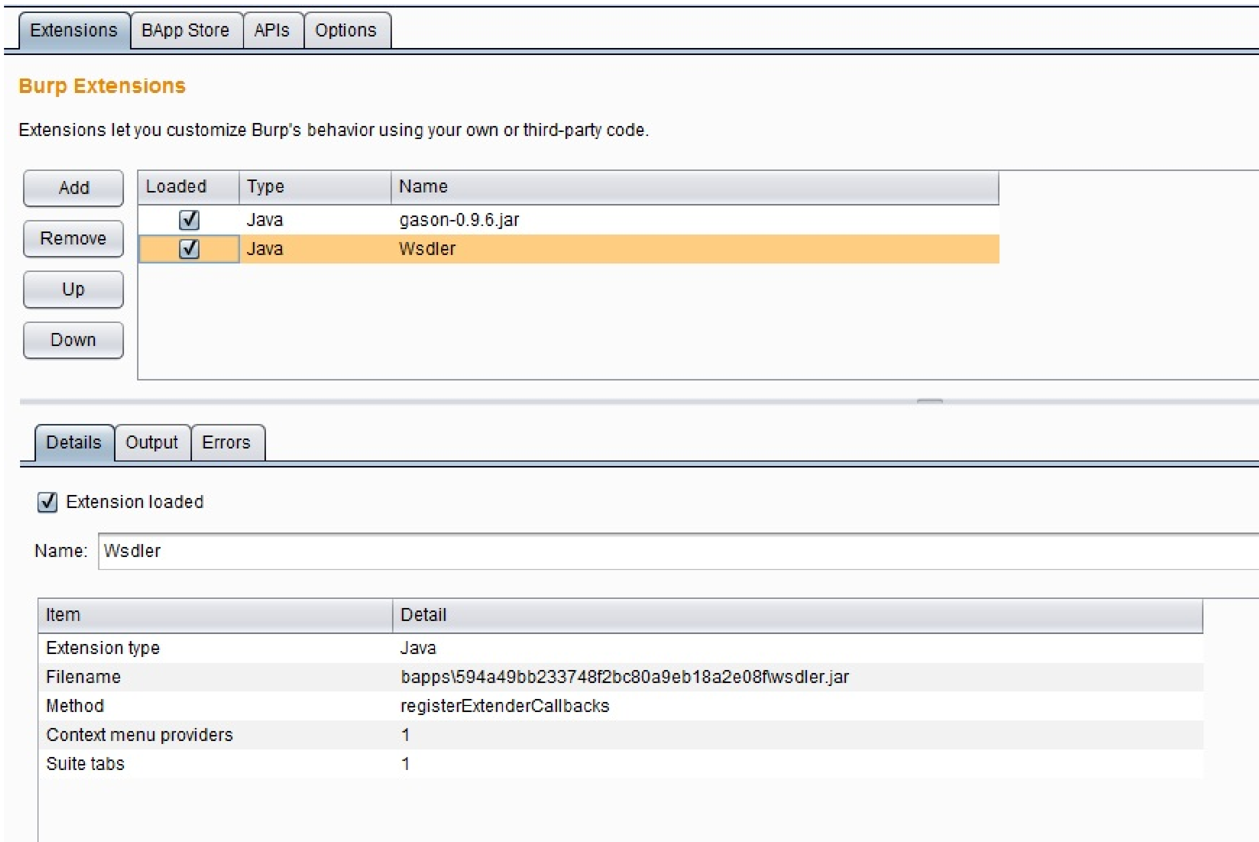
文件类型: 所有文件

打开(O)　取消

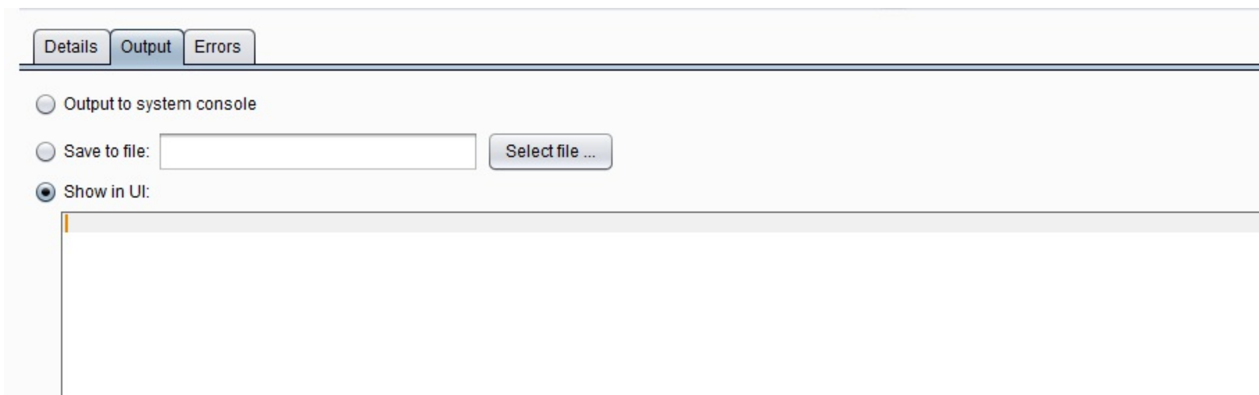
当我们点击图中1处的手工安装按钮，则弹出插件安装文件存储的盘符，选择指定的插件文件，点击打开即可进行安装。

管理和加载Burp 插件（Extension）

从上一章节我们已经了解到，安装完成的插件，都会显示在插件列表中。



如果我们想对某个插件的配置信息进行编辑，则如上图中所示，选中插件，其下方的【Details】标签页会显示插件的拓展信息，如：拓展的插件类型（java/Python/Ruby）、插件的文件名、存储的位置。 除了【Details】标签页外，【Output】和【Errors】两个页面分别可以设置此插件的标准输出和错误信息输出信息。



从上图中我们可以看出，日志信息的输出有三种方式： a)系统控制台输出 b)存储到指定的文件中 c)Burp的界面输出 默认情况下，会选择Burp的界面输出。在实际应用中，我们可以根据自己的需要，对日志的存储方式进行调整。

其他选项设置

Burp插件的其他选项设置主要是指Options 的Tab页中的相关设置。

Burp Intruder Repeater Window Help

Target | Proxy | Spider | Scanner | Intruder | Repeater | Sequencer | Decoder | Comparer | Extender | Options | Alerts | Wsdler

Extensions | BApp Store | APIs | Options

Settings

This setting controls how Burp handles extensions on startup.

☑ Automatically reload extensions on startup

Java Environment

These settings let you configure the environment for executing extensions that are written in Java. If your extensions use any libraries, you can specify a folder fron

Folder for loading library JAR files (optional):

Select folder ...

Python Environment

These settings let you configure the environment for executing extensions that are written in Python. To use Python extensions, you will need to download Jython,

Location of Jython standalone JAR file:

D:\soft\safetool\Burpsuite_pro\jython-installer-2.7.0.jar Select file ...

Folder for loading modules (optional):

Select folder ...

Ruby Environment

These settings let you configure the environment for executing extensions that are written in Ruby. To use Ruby extensions, you will need to download JRuby, whi
can either configure the location of the JRuby JAR file here, or you can load the JAR file on startup via the Java classpath.

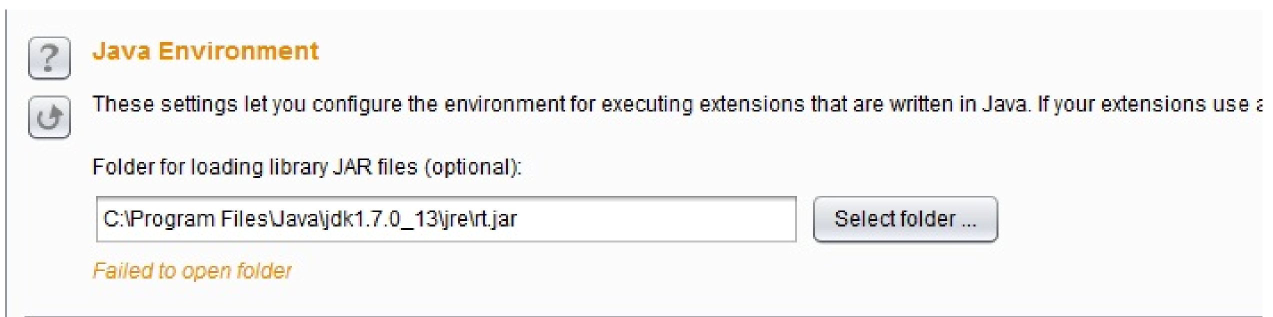
Location of JRuby JAR file:

Select file ...

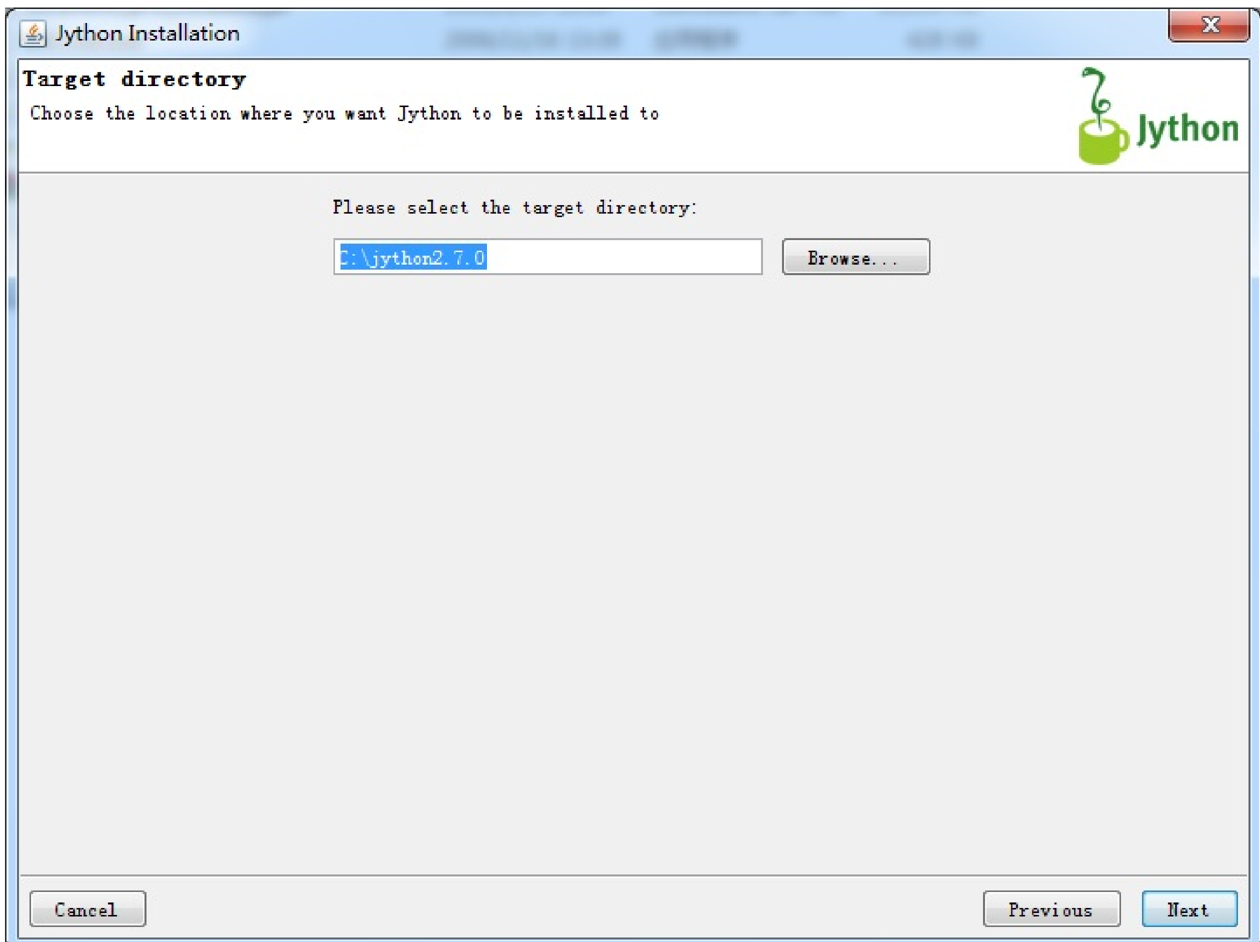
从图中我们可以看出，【Setting】的设置是指：是否启动时自动重新加载burp插件，当我们选择此项时，Burp在重启时，会自动加载Burp在上次关闭时加载的插件内容；而剩下的三项设置是根据插件类型的不同时所需要的运行环境的配置。我们先来看第一个运行环境【Java Environment】。



Burp Suite是基于Java语言开发的软件，通常情况下，当你运行此软件时，系统中的JAVA_HOME、CLASS_PATH、LIB_PATH变量均已正确地配置完成，否则你是难以运行Burp Suite的，所以，通常情况下你是无须再配置此参数；如果实在需要配置，你的插件需要特殊的jdk版本要求或者其他ja，则选择将jar添加即可。



而【Python Environment】和【Ruby Environment】是Burp插件的Python运行环境和Ruby运行环境的配置。前文我们已经知道，Burp是java语言编写的软件，所以运行Python和Ruby需要配置兼容Java与Python、Java与Ruby的jar,默认情况下，Burp支持的为JPython和JRuby，这两个软件的地址分别是 ：http://www.jython.org/、http://jruby.org.其安装方式非常简单，此处以JPython为例： **1.**下载JPython的安装包，Jpython的安装分jython-installer-2.7.0.jar和jython-standalone-2.7.0.jar两个。如果使用jython-installer，则下载完毕后，双击此jar，按照安装向导，一路【Next】到如下图的界面，记录安装路径。然后一直默认，直至安装结束。



如果使用jython-standalone-2.7.0.jar，则直接进行第2步。 **2.**在Burp的Python Environment环境中配置Jpython，如果使用的jython-standalone-2.7.0.jar，则如下图指定jar存放的位置即可；如果是使用jython-installer方式，则指定安装的文件夹，由软件自己加载（此处为了说明使用的方式，两个输入域均输入了，实际使用时，Jpython之输入其中之一即可）。

Python Environment

These settings let you configure the environment for executing extensions that are written in Python. To use Python e

Location of Jython standalone JAR file:

D:\soft\safetool\Burpsuite_pro\jython-standalone-2.7.0.jar Select file ...

Folder for loading modules (optional):

C:\jython2.7.0 Select folder ...

至于JRuby的配置与JPython类似，此处就不再赘述。配置完插件运行的可依赖环境之后，当我们使用插件时就能正常使用，否则，在插件的【Errors】标签页中会有错误的提示信息，我们可以根据错误提示来修改自己的配置。

Extensions | BApp Store | APIs | Options

Burp Extensions

Extensions let you customize Burp's behavior using your own or third-party code.

Add / Remove / Up / Down

Loaded	Type	Name
☑	Java	gason-0.9.6.jar
☑	Java	Wsdler
☑	Python	SQLiPy

Details | Output | Errors

Output to system console

Save to file: Select file ...

Show in UI:

错误提示信息显示

++值得注意的是，当我们使用*Burp*插件功能，对于*Burp*运行时所需要的*JVM*内存占用比较大，一般建议设置为*1G*，具体设置请参考第一章节。++

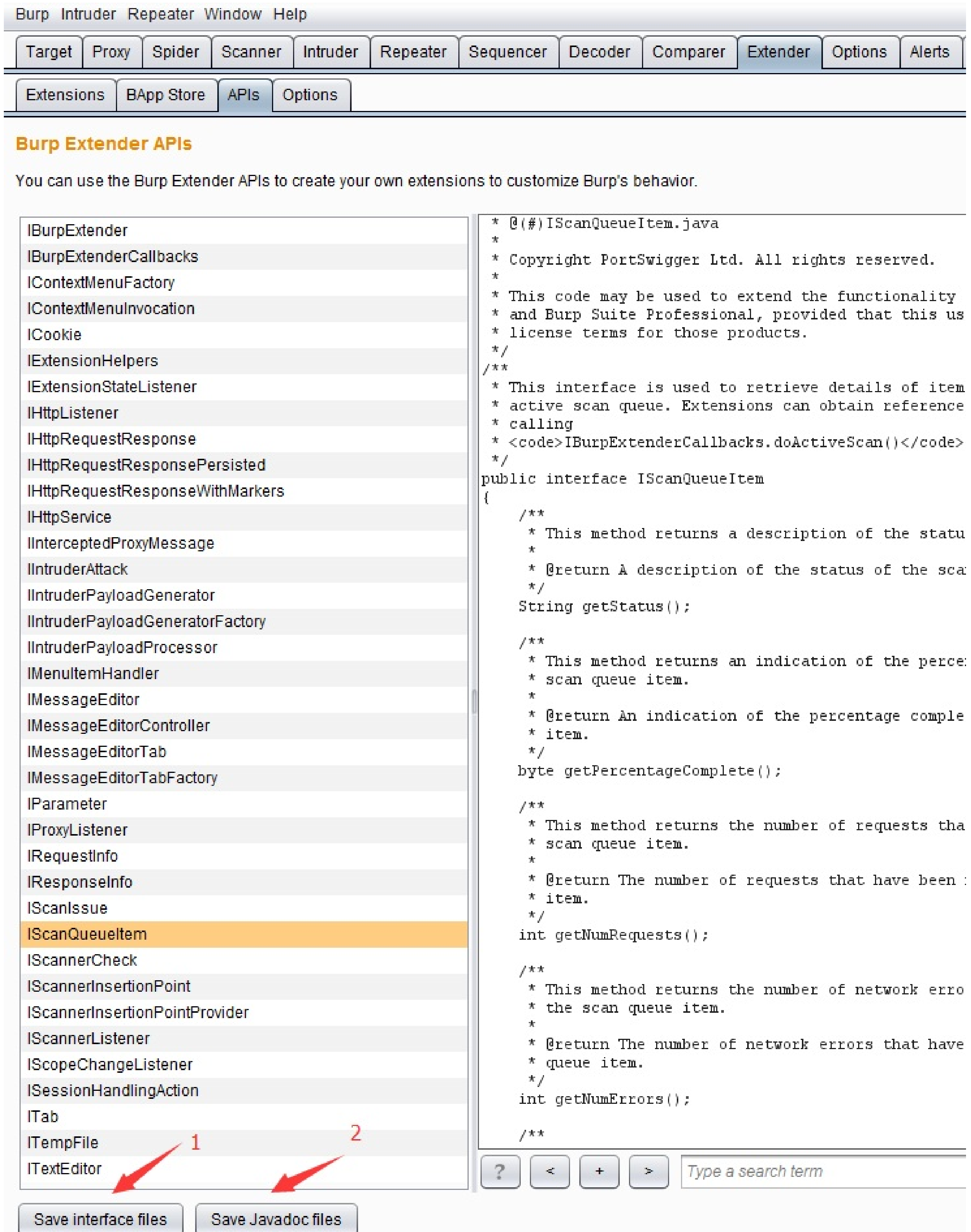
如何编写自己的Burp Suite插件

Burp Suite的强大除了自身提供了丰富的可供测试人员使用的功能外，其提供的支持第三方拓展插件的功能也极大地方便使用者编写自己的自定义插件。从上一章节我们已经了解到，Burp Suite支持的插件类型有Java、Python、Ruby三种。无论哪种语言的实现，开发者只要选择自己熟悉的语言，按照接口规范去实现想要的功能即可。下面我们就来看看如何开发一个Burp Extender的插件。 本章讲述的主要内容有：

- API简述
- Burp插件的编写前准备
- Burp插件的编写（Java语言版）

API简述

打开Burp Extender的APIs的Tab页，看到的界面如下图所示：



界面由左边的接口类和右边的接口定义和描述构成，其中左边的最下端有两个按钮，图中1按钮为保存接口类，当我们点击保存后，在指定的存储目录下，会生成一系列的java文件，如下图：

burp

帮助(H)

打印 新建文件夹

名称	修改日期	类型	大小
IBurpExtender.java	2016/11/14 14:48	UEStudio Docu...	1 KB
IBurpExtenderCallbacks.java	2016/11/14 14:48	UEStudio Docu...	41 KB
IContextMenuFactory.java	2016/11/14 14:48	UEStudio Docu...	2 KB
IContextMenuInvocation.java	2016/11/14 14:48	UEStudio Docu...	6 KB
ICookie.java	2016/11/14 14:48	UEStudio Docu...	2 KB
IExtensionHelpers.java	2016/11/14 14:48	UEStudio Docu...	14 KB
IExtensionStateListener.java	2016/11/14 14:48	UEStudio Docu...	1 KB
IHttpListener.java	2016/11/14 14:48	UEStudio Docu...	2 KB
IHttpRequestResponse.java	2016/11/14 14:48	UEStudio Docu...	3 KB
IHttpRequestResponsePersisted.java	2016/11/14 14:48	UEStudio Docu...	1 KB
IHttpRequestResponseWithMarkers.java	2016/11/14 14:48	UEStudio Docu...	2 KB
IHttpService.java	2016/11/14 14:48	UEStudio Docu...	1 KB
IInterceptedProxyMessage.java	2016/11/14 14:48	UEStudio Docu...	5 KB
IIntruderAttack.java	2016/11/14 14:48	UEStudio Docu...	1 KB
IIntruderPayloadGenerator.java	2016/11/14 14:48	UEStudio Docu...	2 KB
IIntruderPayloadGeneratorFactory.java	2016/11/14 14:48	UEStudio Docu...	2 KB
IIntruderPayloadProcessor.java	2016/11/14 14:48	UEStudio Docu...	2 KB
IMenuItemHandler.java	2016/11/14 14:48	UEStudio Docu...	2 KB
IMessageEditor.java	2016/11/14 14:48	UEStudio Docu...	2 KB
IMessageEditorController.java	2016/11/14 14:48	UEStudio Docu...	2 KB
IMessageEditorTab.java		UEStudio Docu...	4 KB
IMessageEditorTabFactory.java		UEStudio Docu...	2 KB
IParameter.java		UEStudio Docu...	4 KB

类型: UEStudio Document (.java)
大小: 1.91 KB
修改日期: 2016/11/14 14:48

这些文件的内容即为前一张图中右边所示的内容，按照java语言的源文件格式存放的，在编写插件时，可直接将burp包引入Project中使用。而前一张图中2按钮为保存Javadocs,点击保存后，会在存储目录中存放与API相对应的JavaDocs文件。用浏览器打开则如下图所示：

All Classes

IBurpExtender
IBurpExtenderCallbacks
IContextMenuFactory
IContextMenuInvocation
ICookie
IExtensionHelpers
IExtensionStateListener
IHttpListener
IHttpRequestResponse
IHttpRequestResponsePersisted
IHttpRequestResponseWithMarkers
IHttpService
IInterceptedProxyMessage
IIntruderAttack
IIntruderPayloadGenerator
IIntruderPayloadGeneratorFactory
IIntruderPayloadProcessor
IMenuItemHandler
IMessageEditor
IMessageEditorController
IMessageEditorTab
IMessageEditorTabFactory
IParameter
IProxyListener
IRequestInfo
IResponseInfo
IScanIssue
IScannerCheck
IScannerInsertionPoint
IScannerInsertionPointProvider
IScannerListener
IScanQueueItem
IScopeChangeListener
ISessionHandlingAction
ITab
ITempFile
ITextEditor

Package Class Deprecated Index

Prev Package Next Package Frames No Frames

Package burp

Interface Summary

Interface	Description
IBurpExtender	All extensions must implement this interface.
IBurpExtenderCallbacks	This interface is used by Burp Suite to pass to extensions a set of callback methods that can be used by extensions to perform various actions within Burp.
IContextMenuFactory	Extensions can implement this interface and then call IBurpExtenderCallbacks.registerContextMenuFactory() to register a factory for custom context menu items.
IContextMenuInvocation	This interface is used when Burp calls into an extension-provided IContextMenuFactory with details of a context menu invocation.
ICookie	This interface is used to hold details about an HTTP cookie.
IExtensionHelpers	This interface contains a number of helper methods, which extensions can use to assist with various common tasks that arise for Burp extensions.
IExtensionStateListener	Extensions can implement this interface and then call IBurpExtenderCallbacks.registerExtensionStateListener() to register an extension state listener.
IHttpListener	Extensions can implement this interface and then call IBurpExtenderCallbacks.registerHttpListener() to register an HTTP listener.
IHttpRequestResponse	This interface is used to retrieve and update details about HTTP messages.
IHttpRequestResponsePersisted	This interface is used for an IHttpRequestResponse object whose request and response messages have been saved to temporary files using IBurpExtenderCallbacks.saveBuffersToTempFiles().
IHttpRequestResponseWithMarkers	This interface is used for an IHttpRequestResponse object that has had markers applied.
IHttpService	This interface is used to provide details about an HTTP service, to which HTTP requests can be sent.
IInterceptedProxyMessage	This interface is used to represent an HTTP message that has been intercepted by Burp Proxy.
IIntruderAttack	This interface is used to hold details about an Intruder attack.
IIntruderPayloadGenerator	This interface is used for custom Intruder payload generators.
IIntruderPayloadGeneratorFactory	Extensions can implement this interface and then call IBurpExtenderCallbacks.registerIntruderPayloadGeneratorFactory() to register a factory for custom Intruder payloads.
IIntruderPayloadProcessor	Extensions can implement this interface and then call IBurpExtenderCallbacks.registerIntruderPayloadProcessor() to register a custom Intruder payload processor.
IMenuItemHandler	Deprecated *Use IContextMenuFactory instead.*
IMessageEditor	This interface is used to provide extensions with an instance of Burp's HTTP message editor, for the extension to use in its own UI.
IMessageEditorController	This interface is used by an IMessageEditor to obtain details about the currently displayed message.
IMessageEditorTab	Extensions that register an IMessageEditorTabFactory must return instances of this interface, which Burp will use to create custom tabs within its HTTP message editors.
IMessageEditorTabFactory	Extensions can implement this interface and then call IBurpExtenderCallbacks.registerMessageEditorTabFactory() to register a factory for custom message editor tabs.
IParameter	This interface is used to hold details about an HTTP request parameter.

除了上文说的，我们能导出JavaDocs到本地外，Burp官方也提供了一份在线文档，地址为：https://portswigger.net/burp/extender/api/index.html 下面我们根据接口功能的不同对API进行分类。

1. 插件入口和帮助接口类：IBurpExtender、IBurpExtenderCallbacks、IExtensionHelpers、IExtensionStateListener

   > IBurpExtender接口类是Burp插件的入口，所有Burp的插件均需要实现此接口，并且类命名为BurpExtender。 IBurpExtenderCallbacks接口类是IBurpExtender接口的实现类与Burp其他各个组件（Scanner、Intruder、Spider......）、各个通信对象（HttpRequestResponse、HttpService、SessionHandlingAction）之间的纽带。 IExtensionHelpers、IExtensionStateListener这两个接口类是插件的帮助和管理操作的接口定义。

2. UI相关接口类：IContextMenuFactory、IContextMenuInvocation、ITab、ITextEditor、IMessageEditor、IMenuItemHandler

   > 这类接口类主要是定义Burp插件的UI显示和动作的处理事件，主要是软件交互中使用。

3. Burp工具组件接口类：IInterceptedProxyMessage、IIntruderAttack、IIntruderPayloadGenerator、IIntruderPayloadGeneratorFactory、IIntruderPayloadProcessor、IProxyListener、IScanIssue、IScannerCheck、IScannerInsertionPoint、IScannerInsertionPointProvider、IScannerListener、

IScanQueueItem、IScopeChangeListener

> 这些接口类的功能非常好理解，Burp在接口定义的命名中使用了的见名知意的规范，看到接口类的名称，基本就能猜测出来这个接口是适用于哪个工具组件。

4. HTTP消息处理接口类：ICookie、IHttpListener、IHttpRequestResponse、IHttpRequestResponsePersisted、IHttpRequestResponseWithMarkers、IHttpService、IRequestInfo、IParameter、IResponseInfo

> 这些接口的定义主要是围绕HTTP消息通信过程中涉及的Cookie、Request、Response、Parameter几大消息对象，通过对通信消息头、消息体的数据处理，来达到控制HTTP消息传递的目的。

通过对Burp插件 API的功能划分，我们对API的接口有一个初步的认知，知道在使用某个功能时，可以去哪个接口类中寻找相应的接口定义来做自己的实现。例如。我们想显示一个Tab页界面，那么肯定是要实现ITab接口；如果需要对消息进行编辑修改，则需要实现IMessageEditor接口；需要使用payload生成器，则需要实现IIntruderPayloadGenerator接口。通过接口分类后再找具体的接口定义的方法，可以帮助我们在不太熟悉Burp 插件API的情况下，更快地开发出自己需要的插件。

Burp插件的编写前准备

编写一个完整的Burp插件的大体过程可分为如下三步：

1. 导入**Burp**插件接口，即通过APIs界面上的【save interface files】的保存动作，将生成的文件连同burp目录一下添加你自己的Java Project中。

2. 编写**Burp**插件，即通过自己的代码编写，完成自己想实现的功能插件的编码过程。

3. 加载**Burp**插件，即将上一步编写完成的插件，打包后导入Burp Extensions中，进行试用测试的过程。

其中第一步和第三步对大多数来说，没有难度，主要难度在于如何编码实现Burp的插件。在Burp Suite的官方网站上，插件编写网址：https://portswigger.net/burp/extender/。当我们打开这个网页，会发现网站上有一系列Demo，包含各个编程语言的实现的源代码，这些

Demo，按照开发的难度逐步增加的，我们可以点击【Download】链接下载源码进行分析和学习（网页截图如下所示）。

- **Hello world** - This is a very simple extension that prints some output to various locations within Burp.
  Details Download [Java, Python, Ruby]
- **Event listeners** - This extension registers listeners for various runtime events, and prints a message when each event occurs.
  Details Download [Java, Python, Ruby]
- **Traffic redirector** - This extension redirects all outbound requests from one host to another.
  Details Download [Java, Python, Ruby]
- **Custom logger** - This extension adds a new tab to Burp's user interface, and displays a log of HTTP traffic for all Burp tools, in the style of Burp's Proxy history.
  Details Download [Java, Python, Ruby]
- **Custom editor tab** - This extension adds a new tab to Burp's HTTP message editor, in order to handle an unsupported data serialization format.
  Details Download [Java, Python]
- **Custom scan insertion points** - This extension provides custom attack insertion points for active scanning, allowing Burp's scanning engine to work with an unsupported data serialization format.
  Details Download [Java, Python]
- **Custom scanner checks** - This extension implements custom checks to extend the capabilities of Burp's active and passive scanning engines.
  Details Download [Java]
- **Intruder payloads** - This extension provides custom Intruder payloads and payload processing.
  Details Download [Java]

除了这些Demo外，网站还有一篇插件编写入门的文章。网址：http://blog.portswigger.net/2012/12/writing-your-first-burp-extension.html。文章中以Java和Python语言为例，编写一个最简单的Burp插件来熟悉插件的编写流程，阅读这些文章，会给我们编写Burp插件带来极大的帮助。阅读完这篇文章之后，接着官方的归档文件中，会有一些由浅入深讲解插件编写的文章，E文好的同学也可以自己看看，网址点击：http://blog.portswigger.net/2012_12_01_archive.html

如果你没法读懂这些文章，那么我们一起先来看看编写Burp插件的准备工作有哪些，下一章以实例学习如何编写一个Burp插件。通常编写Burp插件的准备工作有：

1. 安装JDK------我相信会使用Burp Suite软件的同学都已经安装过JDK了，如果没有安装，请阅读此书的第一章第二章相关章节。
2. 安装IDE------一款好的IDE能使得开发效率得到极大的提升，Java语言推荐使用Eclipse或者IntelliJ，Python推荐使用Pycharm或者PyDev，具体每一个IDE软件的安装，请读者自己查找学习。
3. 熟悉编程语言的语法-----这是编写插件的基础，如果连基本的语法都不熟悉，编写Burp代码是有一定难度的，接下来的文章中，编者默认为阅读者对语法的掌握程度是熟悉的。

具备了以上三点，把你自己想要实现的插件功能按照软件需求分析的流程在图纸上简单地画出来，我们即可以进入插件开发环节。

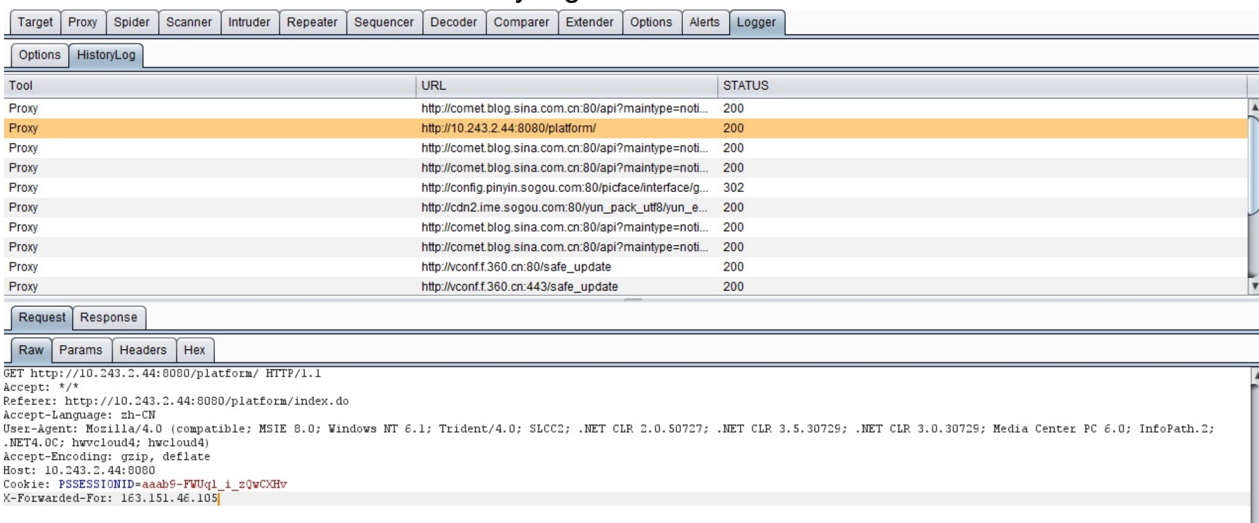
Burp插件的编写（Java语言版）

Burp插件的编写语言有Java、Python、Ruby，此处我们以Java为例，来学习编写一个插件。插件要实现的功能是：在http和https请求的header部分添加一个X-Forward-For字段，而字段中的IP地址是随机生成或者指定的，用于绕过使用该字段来防护暴力破解等的场景。插件代码的编写是基于网友bit4woo的Burp插件源码进行二次开发的。源项目github地址：https://github.com/bit4woo/Burp_Extender_random_X-Forward-For，在此向网友bit4woo致谢！

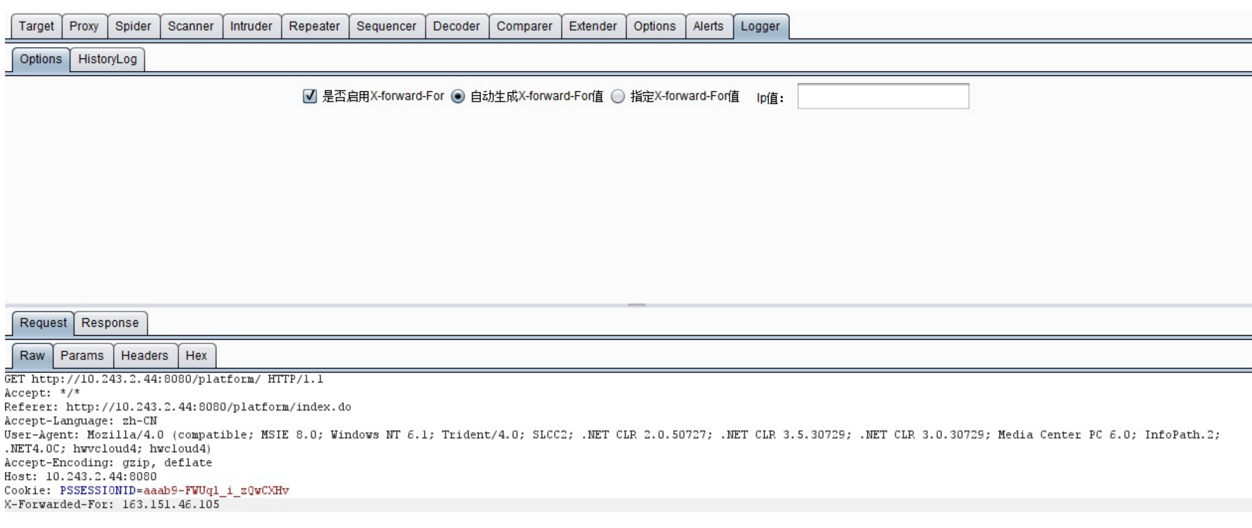
bit4woo网友的源码中实现的插件中仅有X-Forward-For的消息头添加，无插件的UI界面，我们无控制插件是否生效和跟踪http消息通信的直观查看。因此，我们需要实现的插件的功能如下：

1. 对使用插件的HTTP请求消息头中添加X-Forward-For字段
2. 添加UI界面，直观地感受插件的使用。
3. 跟踪HTTP消息，在Burp中使用了哪些组件，请求的URL是什么,请求后的http状态码是否多少。
4. 能在插件中控制本插件是否拦截所有的HTTP请求消息，即是否对请求消息头添加X-Forward-For字段。
5. 添加的X-Forward-For字段是随机生成还是自己指定的值。

插件编写完成的消息跟踪界面（HistoryLog）如下图：

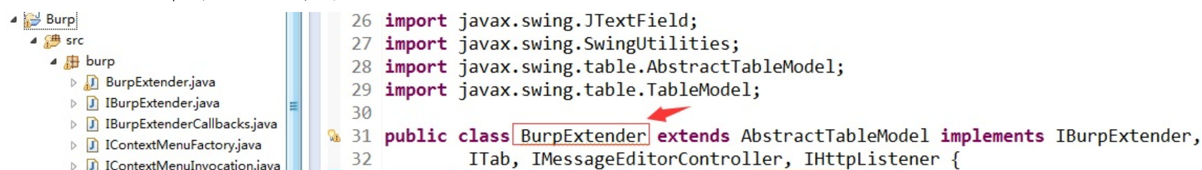


插件的设置界面（Options）如下：



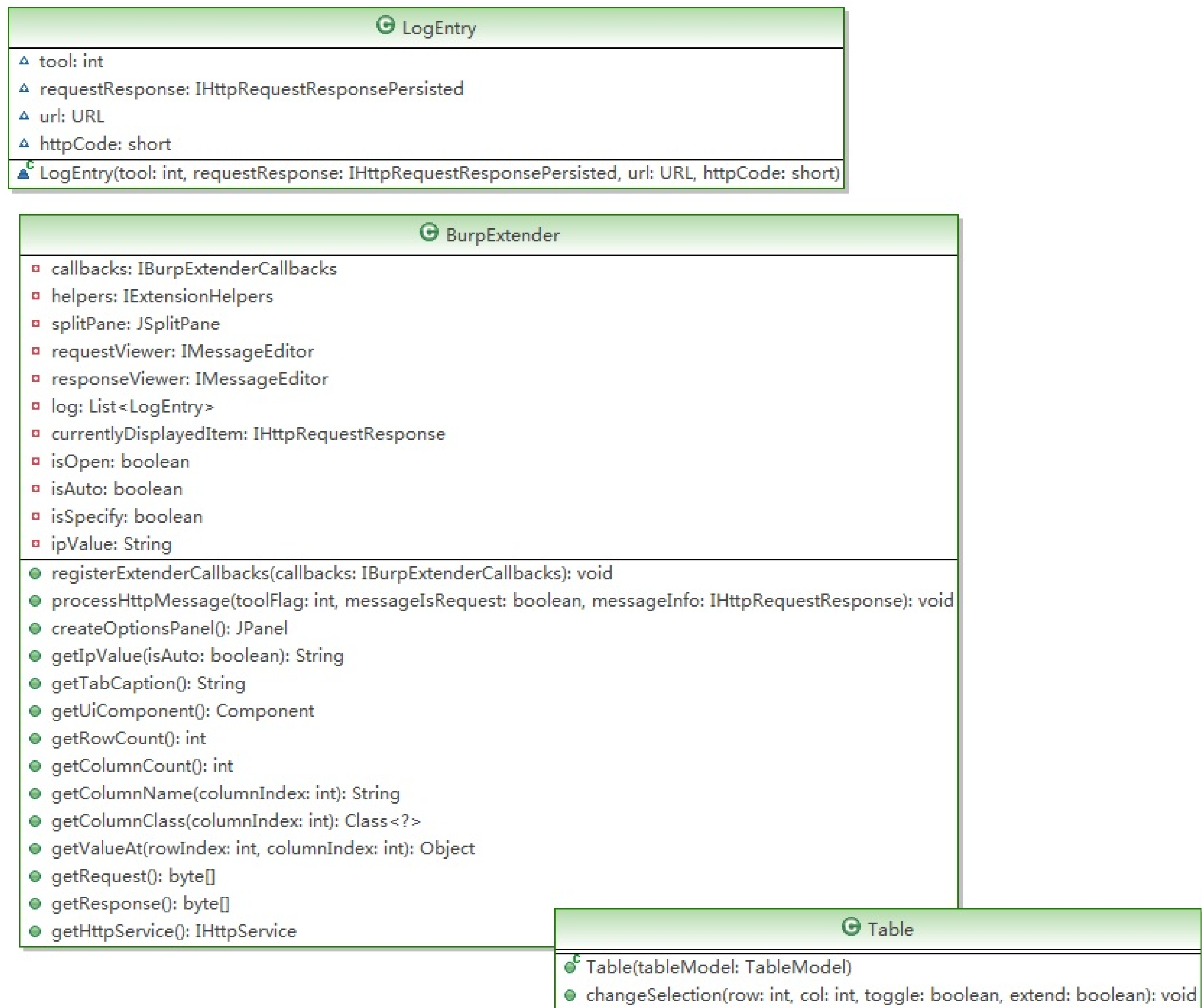
下面我们就来看看具体的编码实现（此处仅仅谈Burp插件的编写，Swing组件的使用不涉及，默认编写者对Swing已熟练掌握）。

1.首先在burp包中定义了一个名称为BurpExtender的java类，必须继承IBurpExtender接口。这个上一个章节已经阐述过了。



2.因为要在Burp中添加一个tab页作为我们自定义的UI，所以我们需要实现ITab接口；因为要显示请求和响应消息，所以需要实现IMessageEditorController接口；因为要拦截请求的报文，添加X-Forward-For，所以需要实现IHttpListener接口。如上图所示。类定义完成后，导入未实现的方法，则类的UML图如下：

2.因为要在Burp中添加一个tab页作为我们自定义的UI，所以我们需要实现ITab接口；因为要显示请求和响应消息，所以需要实现IMessageEditorController接口；因为要拦截请求的报文，添加X-Forward-For，所以需要实现IHttpListener接口。如上图所示。类定义完成后，导入未实现的方法，则类的UML图如下：



3.接着就是对接口类的方法实现，在UML中，下面两个是需要实现的主要函数：

> **registerExtenderCallbacks(final IBurpExtenderCallbacks callbacks)** 这个函数是Burp插件的入口，在这里主要做了如下工作：1）初始化插件和组件对象 2）设置自定义的UI界面原型。

```java
@Override
public void registerExtenderCallbacks(final IBurpExtenderCallbacks callbacks) {
    this.callbacks = callbacks;
    helpers = callbacks.getHelpers();
    callbacks.setExtensionName("Random X-forward-For"); // 插件名称
    // 开始创建自定义UI
    SwingUtilities.invokeLater(new Runnable() {
        public void run() {
    });
}
```

> 其中创建自定义UI的**run**函数代码如下：

```
// 主面板
splitPane = new JSplitPane(JSplitPane.VERTICAL_SPLIT);
JTabbedPane topTabs = new JTabbedPane();
// HistoryLog 视图
Table logTable = new Table(BurpExtender.this);
JScrollPane scrollPane = new JScrollPane(logTable);
// 创建【options】显示面板
JPanel optionsPanel = BurpExtender.this.createOptionsPanel();

// 添加主面板的上半部分中，分两个tab页
topTabs.add("Options", optionsPanel);
topTabs.add("HistoryLog", scrollPane);
splitPane.setLeftComponent(topTabs);

// request/response 视图
JTabbedPane tabs = new JTabbedPane();
requestViewer = callbacks.createMessageEditor(
        BurpExtender.this, false);
responseViewer = callbacks.createMessageEditor(
        BurpExtender.this, false);

// 添加主面板的下半部分中，分两个tab页
tabs.addTab("Request", requestViewer.getComponent());
tabs.addTab("Response", responseViewer.getComponent());
splitPane.setRightComponent(tabs);

// 自定义自己的组件
callbacks.customizeUiComponent(splitPane);
callbacks.customizeUiComponent(topTabs);
callbacks.customizeUiComponent(tabs);

// 在Burp添加自定义插件的tab页
callbacks.addSuiteTab(BurpExtender.this);

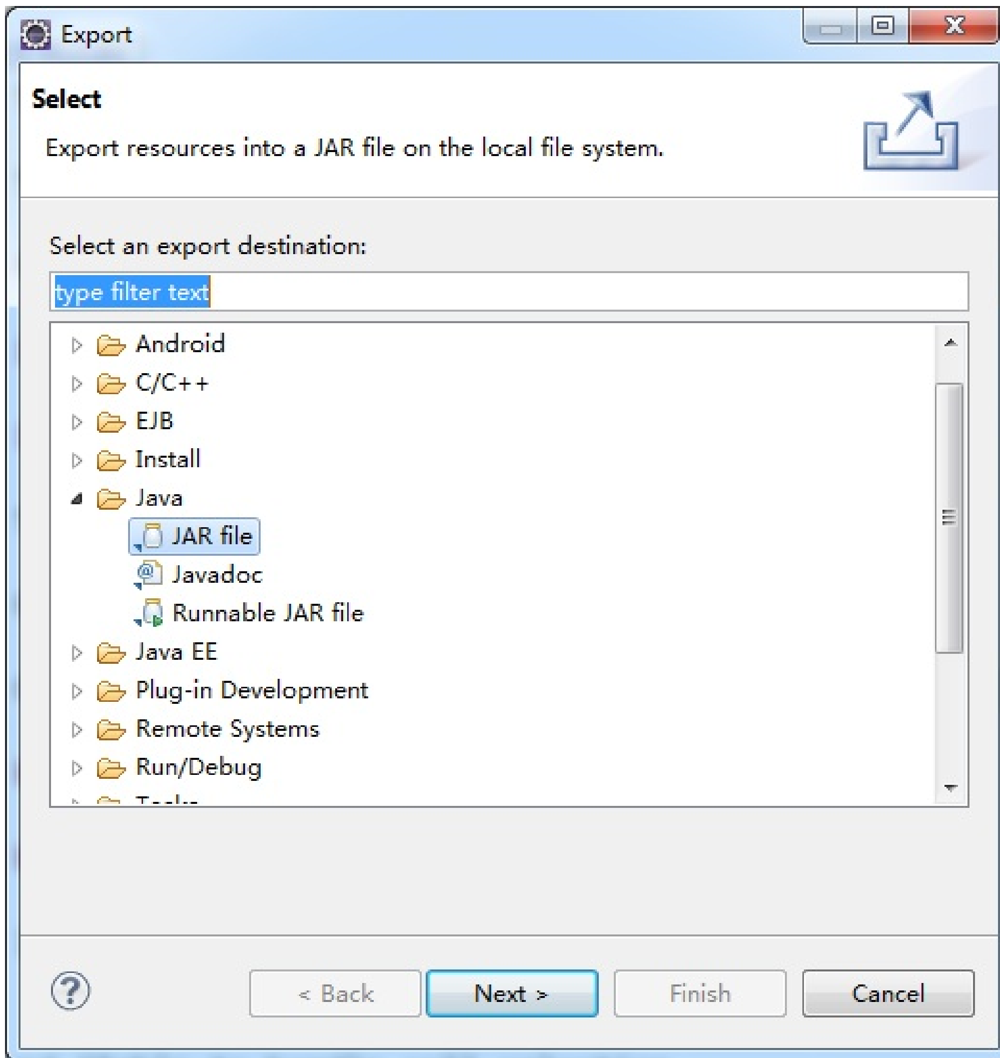
// 注册HTTP listener
callbacks.registerHttpListener(BurpExtender.this);
```

其次是**processHttpMessage(int toolFlag, boolean messageIsRequest, IHttpRequestResponse messageInfo)** 这个函数的功能主要是对HTTP消息的处理和添加HTTP消息到History列表中。其代码如下：

```
public void processHttpMessage(int toolFlag, boolean messageIsRequest,
        IHttpRequestResponse messageInfo) {
    //如果插件未启用，则跳出不执行
    if (!isOpen)return;
    try {
        // 不同的toolflag代表了不同的burp组件，如INTRUDER,SCANNER,PROXY,SPIDER
        if (toolFlag == callbacks.TOOL_PROXY || toolFlag == callbacks.TOOL_INTRUDER
                || toolFlag == callbacks.TOOL_SCANNER || toolFlag == callbacks.TOOL_SPIDER) {
            if (messageIsRequest) { // 对请求包进行处理
                IRequestInfo analyzeRequest = helpers
                        .analyzeRequest(messageInfo); // 对消息体进行解析
                String request = new String(messageInfo.getRequest());
                byte[] body = request.substring(
                        analyzeRequest.getBodyOffset()).getBytes();
                //获取http请求头的信息，返回headers参数的列表
                List<String> headers = analyzeRequest.getHeaders();
                //根据IP生成方式,获取IP值
                String ip ;
                if(isAuto)
                    ip= this.getIpValue(true);
                else
                    ip = this.getIpValue(false);
                String xforward = "X-Forwarded-For: "+ ip;
                //添加X-Forwarded-For
                headers.add(xforward);
                //重新组装请求消息
                byte[] newRequest = helpers.buildHttpMessage(headers, body);
                messageInfo.setRequest(newRequest);// 设置最终新的请求包
            }
            //添加消息到HistoryLog记录中，供UI显示用
            synchronized (log) {
                int row = log.size();
                short httpcode = helpers.analyzeResponse(
                        messageInfo.getResponse()).getStatusCode();
                log.add(new LogEntry(toolFlag, callbacks
                        .saveBuffersToTempFiles(messageInfo), helpers
```

除了这两个函数，其他函数的功能主要是为了UI展示做的各种逻辑操作，此处就不再叙述了，想要了解的同学可以下载本章后面附的源码进行阅读。

4.完成了主要函数的编码之后，插件开发的部分就已经结束了，这时候，我们只需要把代码导出成jar把，加载到Burp Extensions中测试运行即可。



5.本插件和其源码下载地址

点击下载插件jar

点击下载源码

下载完毕后，你可以把src中的两个java类放入从APIs 标签页中导入的接口类所在的burp包中，编译后打包jar运行；也可以直接把下载的X-forward-For.jar导入Burp拓展插件中，即可看到插件的运行界面。

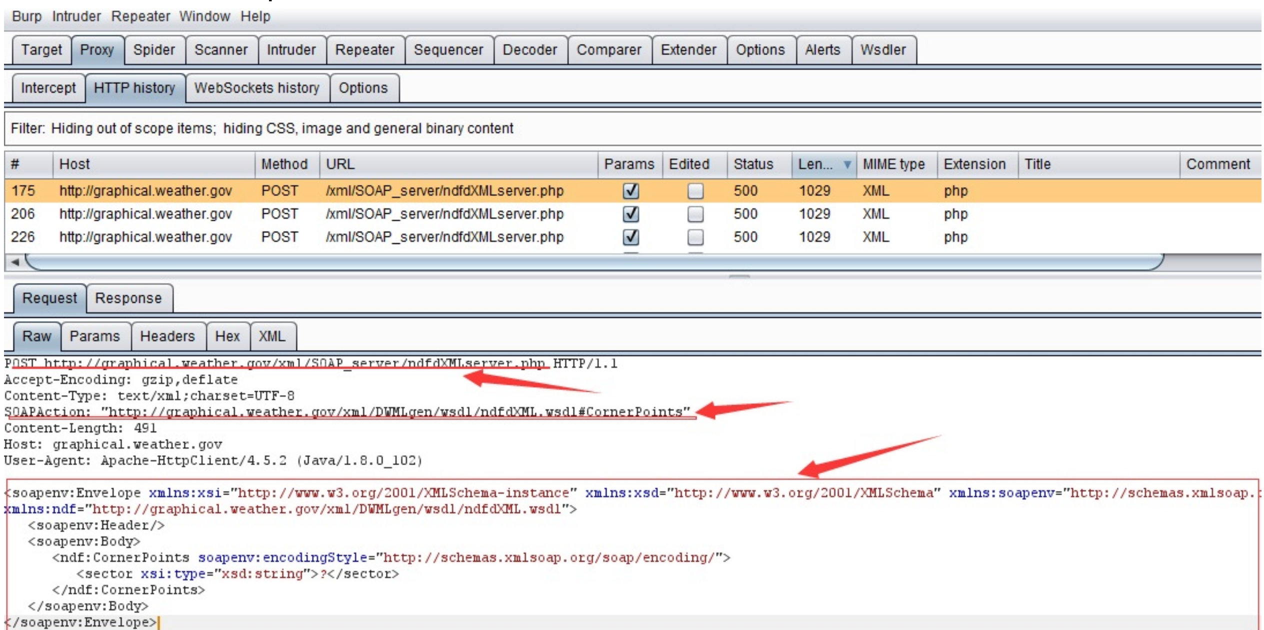
使用Burp Suite测试Web Services服务

从这一章开始，我们进入了Burp的综合使用。通过一系列的使用场景的简单学习，逐渐熟悉Burp在渗透测试中，如何结合其他的工具，组合使用，提高工作效率。本章主要讲述在测试Web Services服务中，如何使用Burp Suite和SoapUI NG Pro的组合，对服务接口进行安全测试。 本章讲述的主要内容有：

- 使用场景和渗透测试环境配置
- 渗透测试过程中组合软件的使用

使用场景和渗透测试环境配置

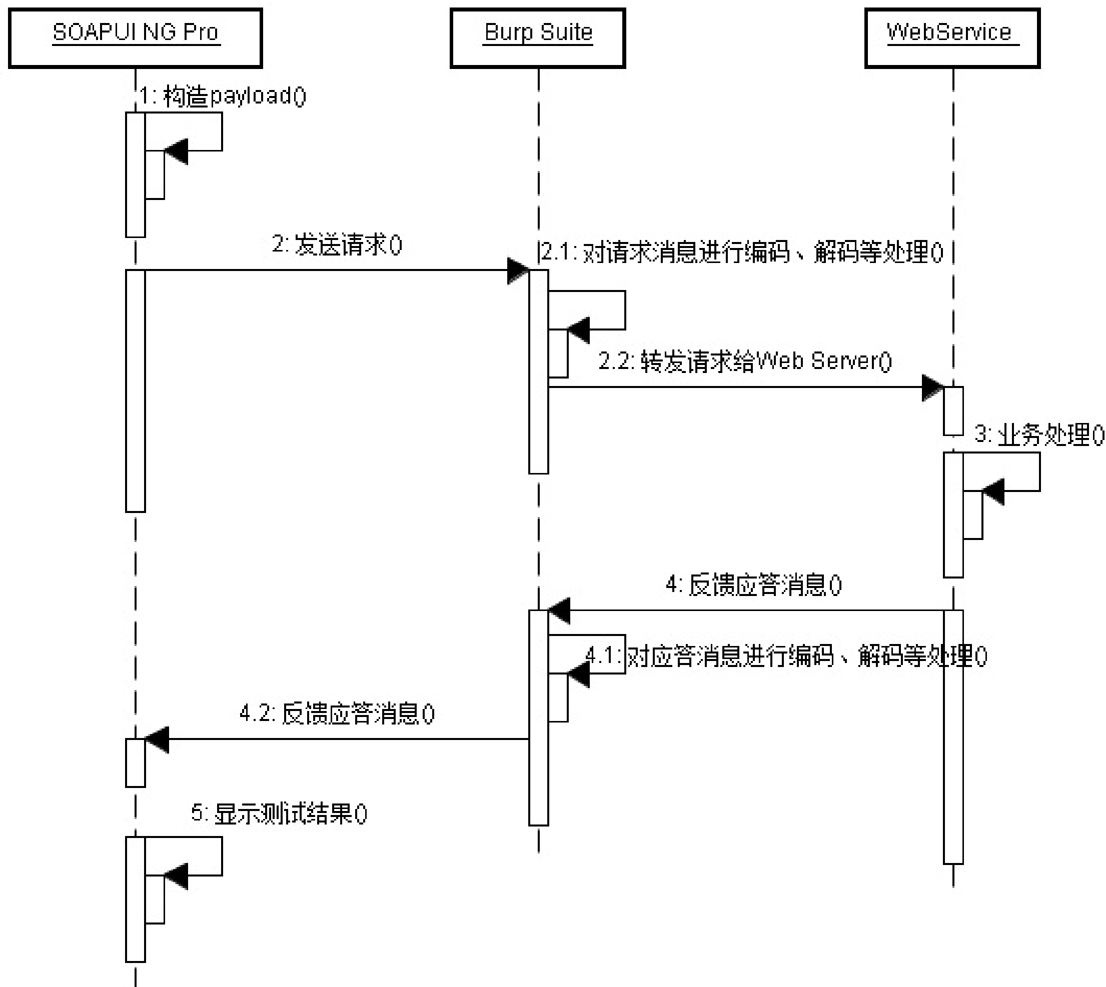
在日常的web测试过程中，除了基于浏览器展现技术的客户端应用程序外，基于SOAP协议进行通信的WebService服务也很常见。WebService的出现是为了解决分布式、跨平台、低耦合而实现的不同编程语言之间采用统一的数据通信的技术规范，在应用程序中，常作为独立的业务模块对外提供具体的业务功能或者为前段提供数据处理的业务接口。因SAOP协议中的接口定义使用XML作为描述性语言，这与php、jsp之类的通信交互在渗透测试上还是有很大的差异。如果使用Burp 对通信消息进行拦截抓包，一次典型的消息内容如下图所示：



其http消息头中包含SOAPAction字段，且消息体为 `<soapenv:Envelope>` 封装的xml文本（更多关于WebService的文章请阅读者自行搜索）。正因为WebService这些特征，所以在渗透测试中我们也需要选择能解析SOAP协议和WSDL描述的软件。这里，我们使用的是SoapUI NG Pro 和Burp Suite。他们各自的作用分别是：

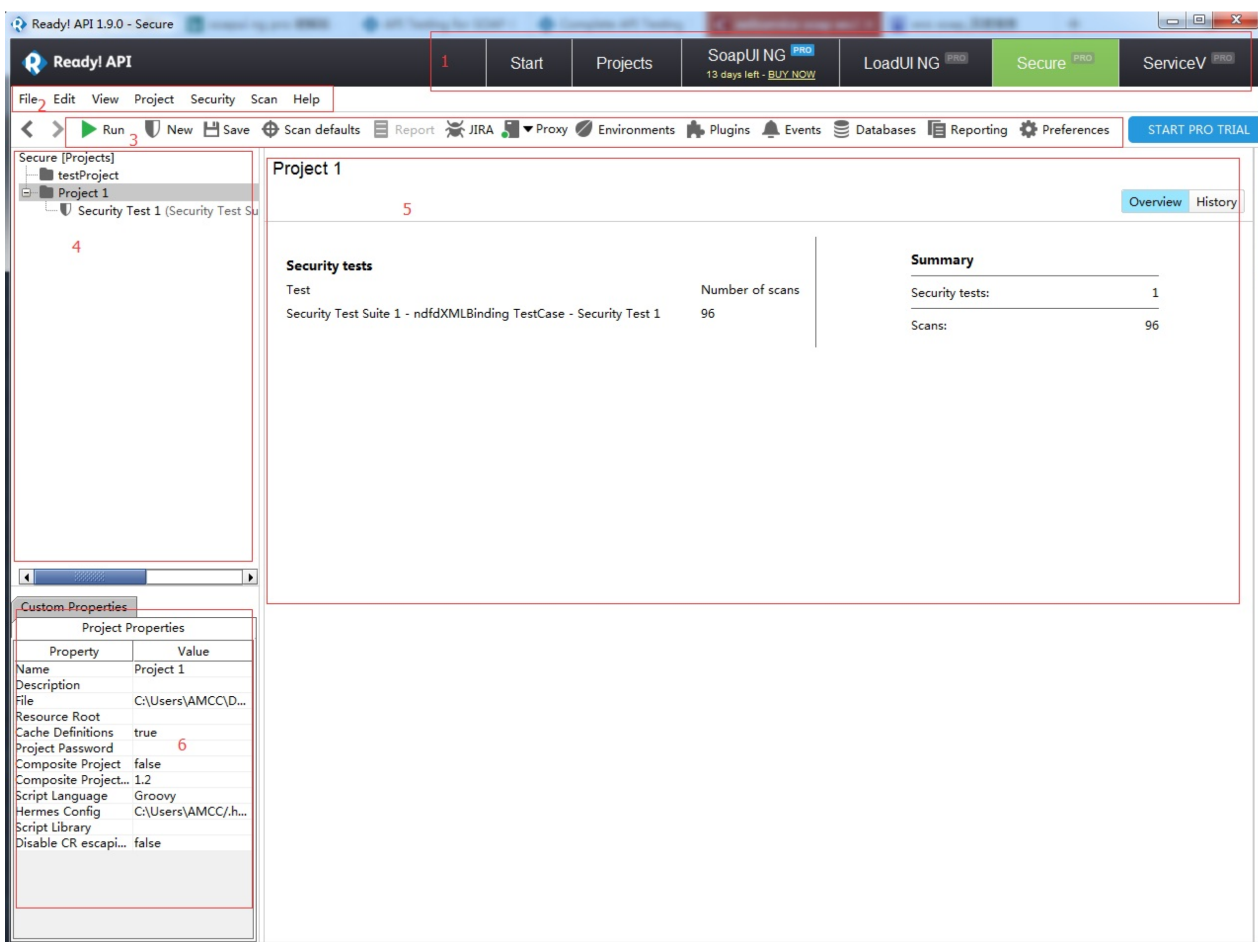
- **SoapUI NG Pro**： 渗透测试流程的发起，通信报文的解析、集合payload之后通信报文的重新组装等。
- **Burp Suite**： 代理拦截，跟踪通信过程和结果，对通信进行重放和二次处理等。

如果按照时序图来展现，他们在通信过程中，各自的时序位置如下：



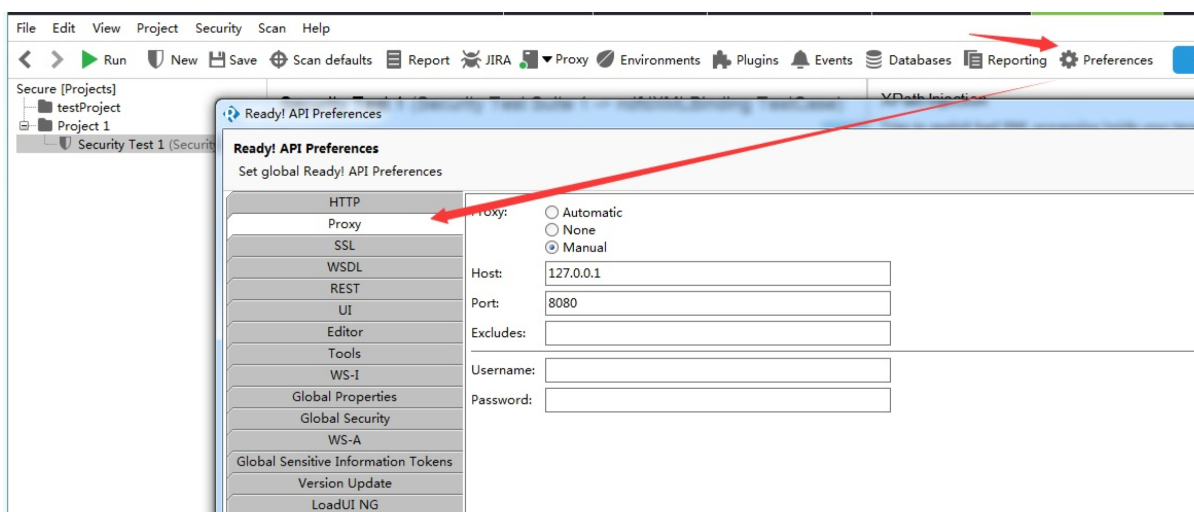
从图中我们可以看出，作为代理服务Burp起着通信中间人的作用，可以对消息进行拦截后的编码、解码、转发、丢弃等各种操作，并记录原始消息。而SoapUI NG Pro作为WebService的测试工具，通过构造不同类型的payload来测试、验证漏洞的存在。他们组合在一起，共同完成复杂场景下WebService服务的渗透测试过程中的安全性验证。

SoapUI NG Pro 是SmartBear公司继SoapUI Pro之后推出的企业应用级收费软件，其试用版下载地址为：https://smartbear.com/product/ready-api/soapui-ng/free-trial/。下载安装完毕后，打开软件的主界面大体如下图所示（其中图中1部分为不同功能视图之间的切换项，图中2部分为菜单栏，图中3部分为常用功能菜单，图中4为Project视图区，图中5为主工作区，图中6部分为属性设置区）：



安装完毕后，我们首先要做的是将SoapUI NG Pro的代理服务指向Burp Suite。假设我的Burp Proxy设置为127.0.0.1:8080。则SoapUI NG Pro的配置是：

1. 点击上图中3部分的**Preferences**，或者上图中2部分的【**File**】>>【**Preferences**】
2. 在弹出的界面中打开**proxy**选项卡，录入代理地址和端口。

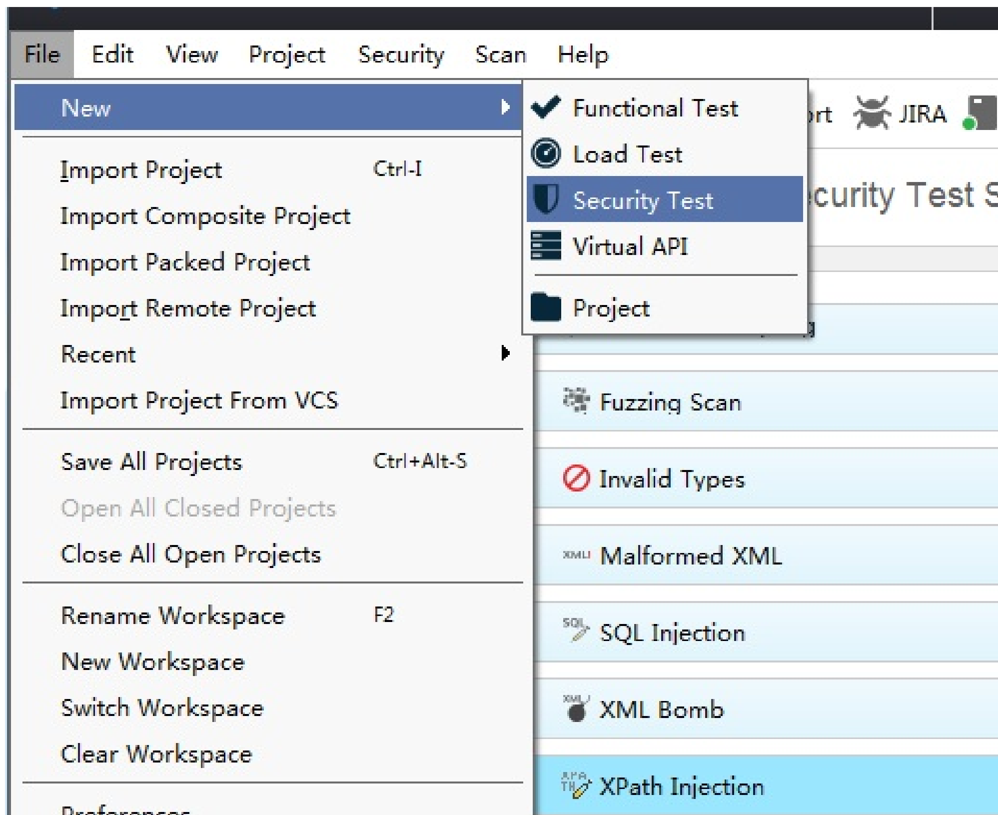


完成以上的配置后，我们对WebService的渗透测试环境已经基本具备，可以开始对一个具体的WebService服务进行渗透测试了。

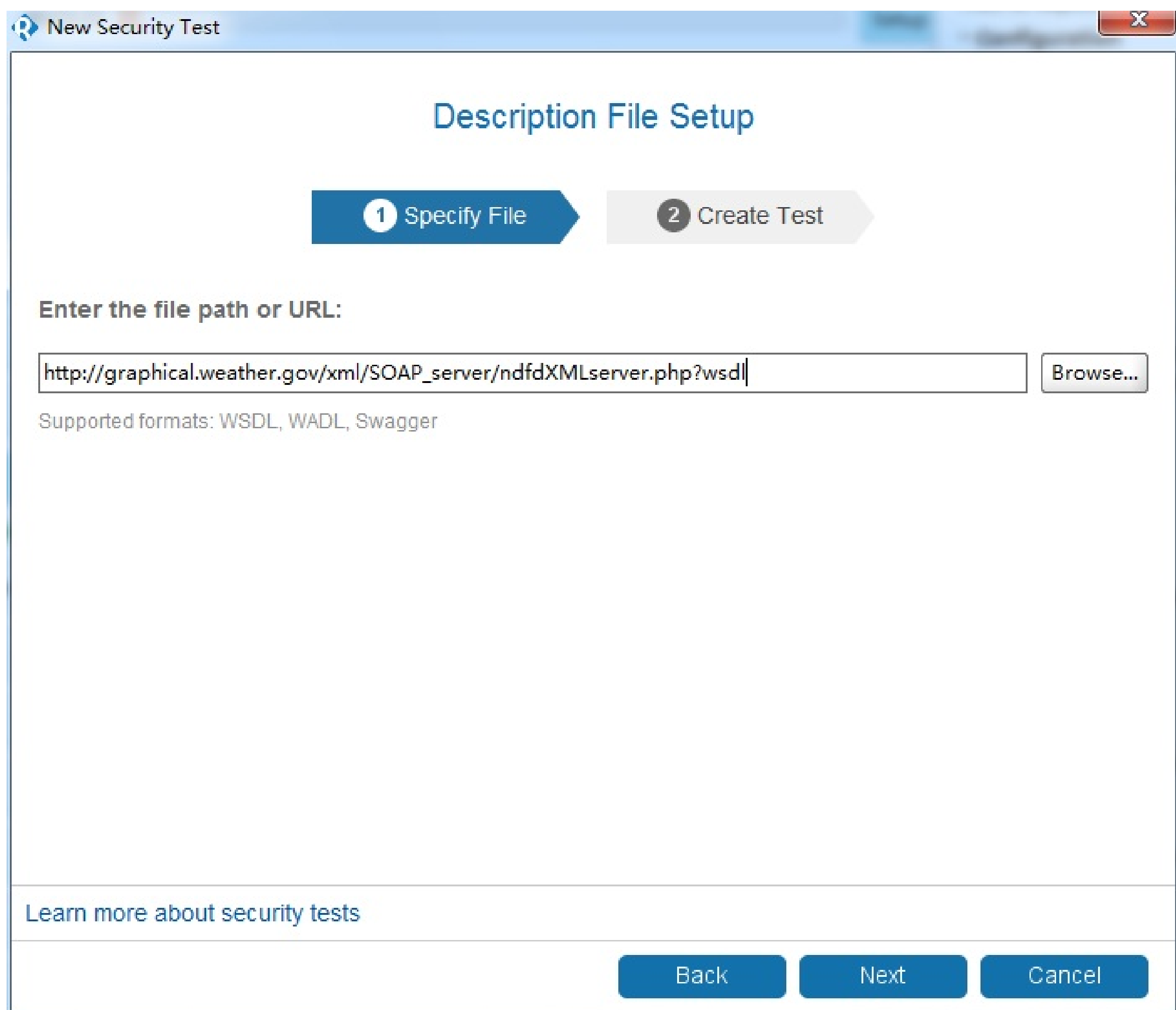
渗透测试过程中组合软件的使用

渗透测试环境配置后，我们就可以开始测试。这里我们可以自己编写WebService服务端，也可以通过搜索引擎选择互联网上公开的WebService，我这里使用的是：http://graphical.weather.gov/xml/SOAP_server/ndfdXMLserver.php?wsdl

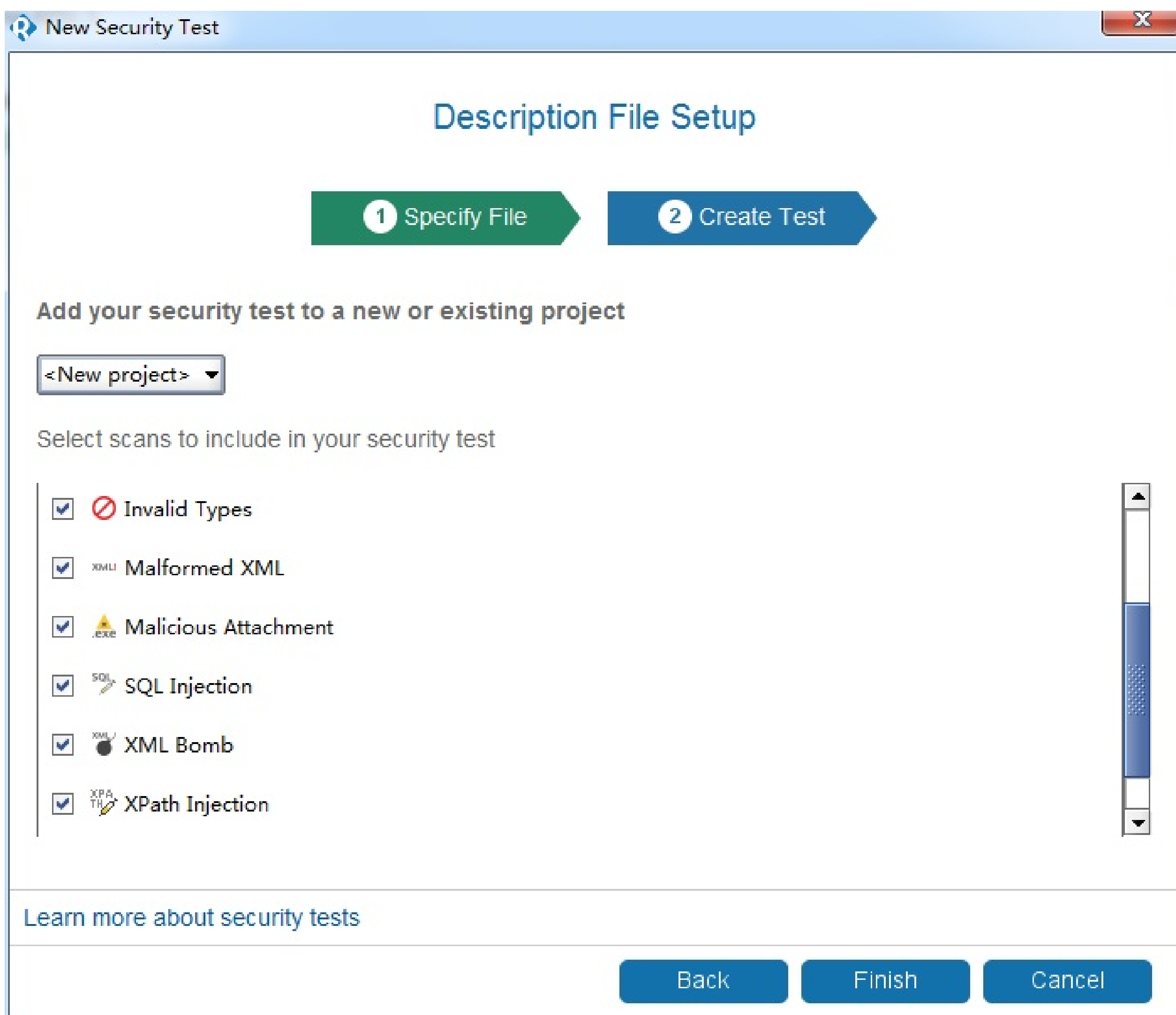
一次简单的渗透测试过程大体包含如下环节： 1.首先，我们通过SoapUI NG Pro 创建安全测试用例。如下图：



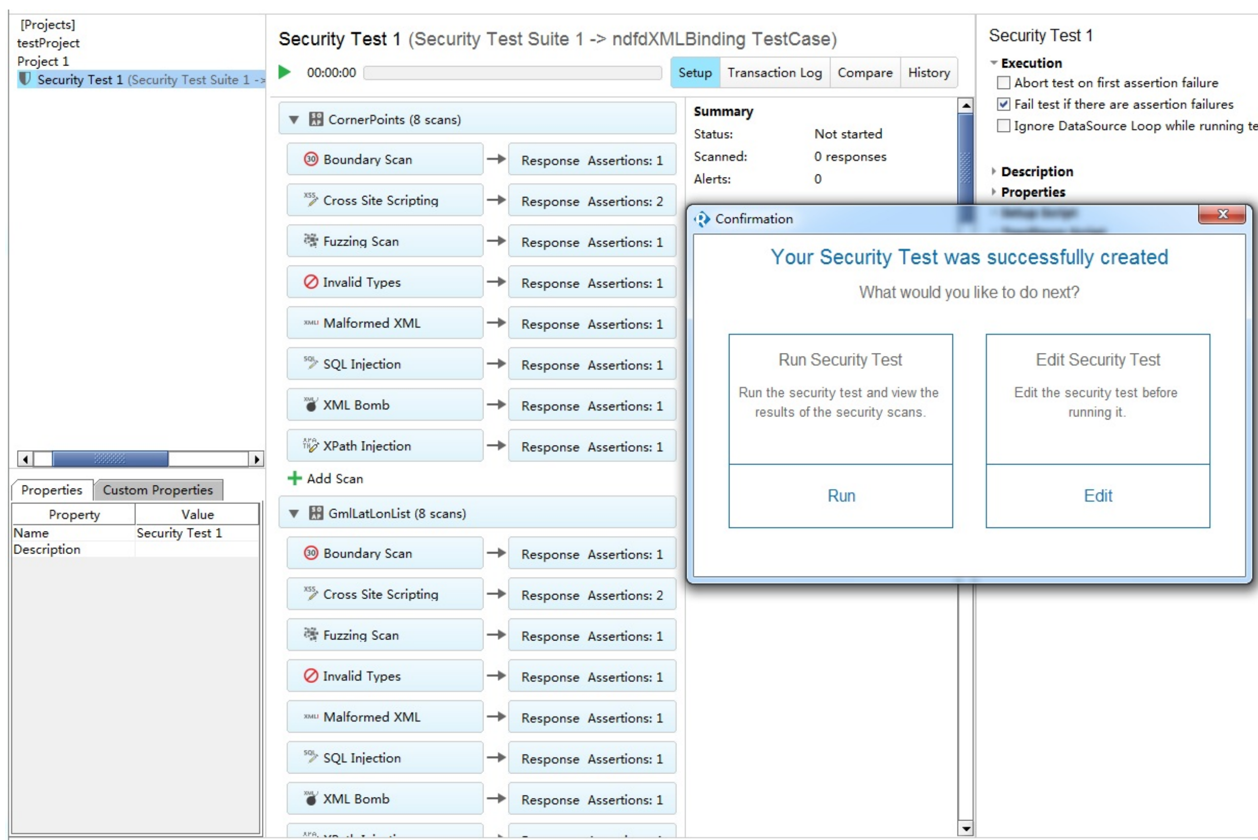
2.在弹出的界面中，选择通过WSDL创建，接着输入WSDL地址。如下图：



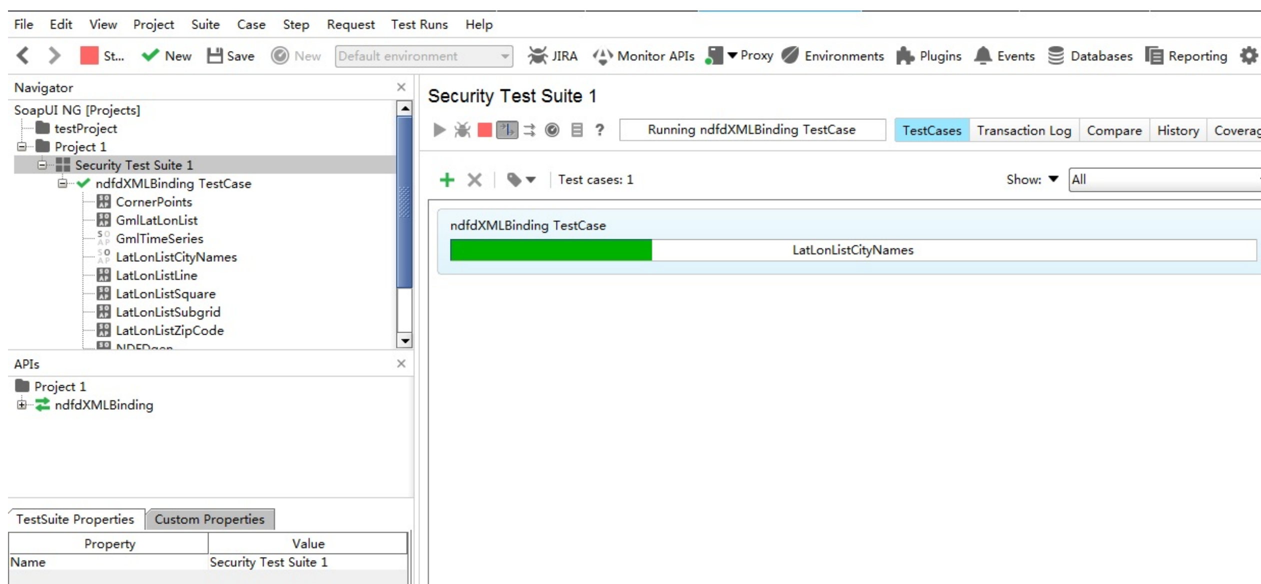
3.当SoapUI NG Pro对WSDL解析完成后，会自动生成一系列的安全测试项：



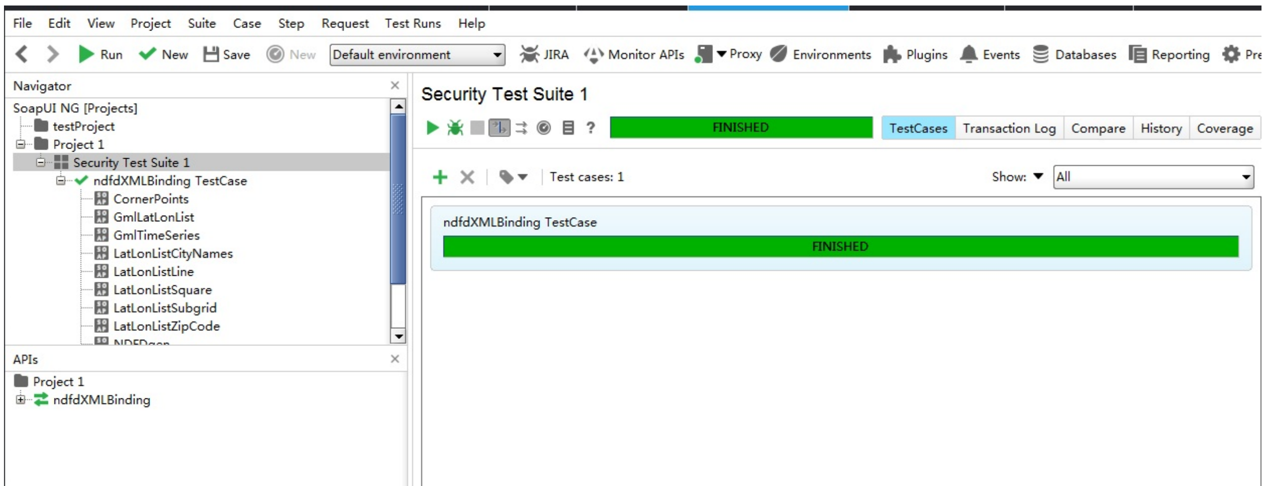
4.我们可以对上图中的安全测试项进行增加和删除，默认情况下，这些安全测试项都是选中的。比如，如果我们只需要测试是否存在XPath注入，则只要上图中的勾选最下面的一项即可。 当SoapUI NG Pro根据安全测试项，完成不同的测试用例的创建之后，主操作界面如下图所示：



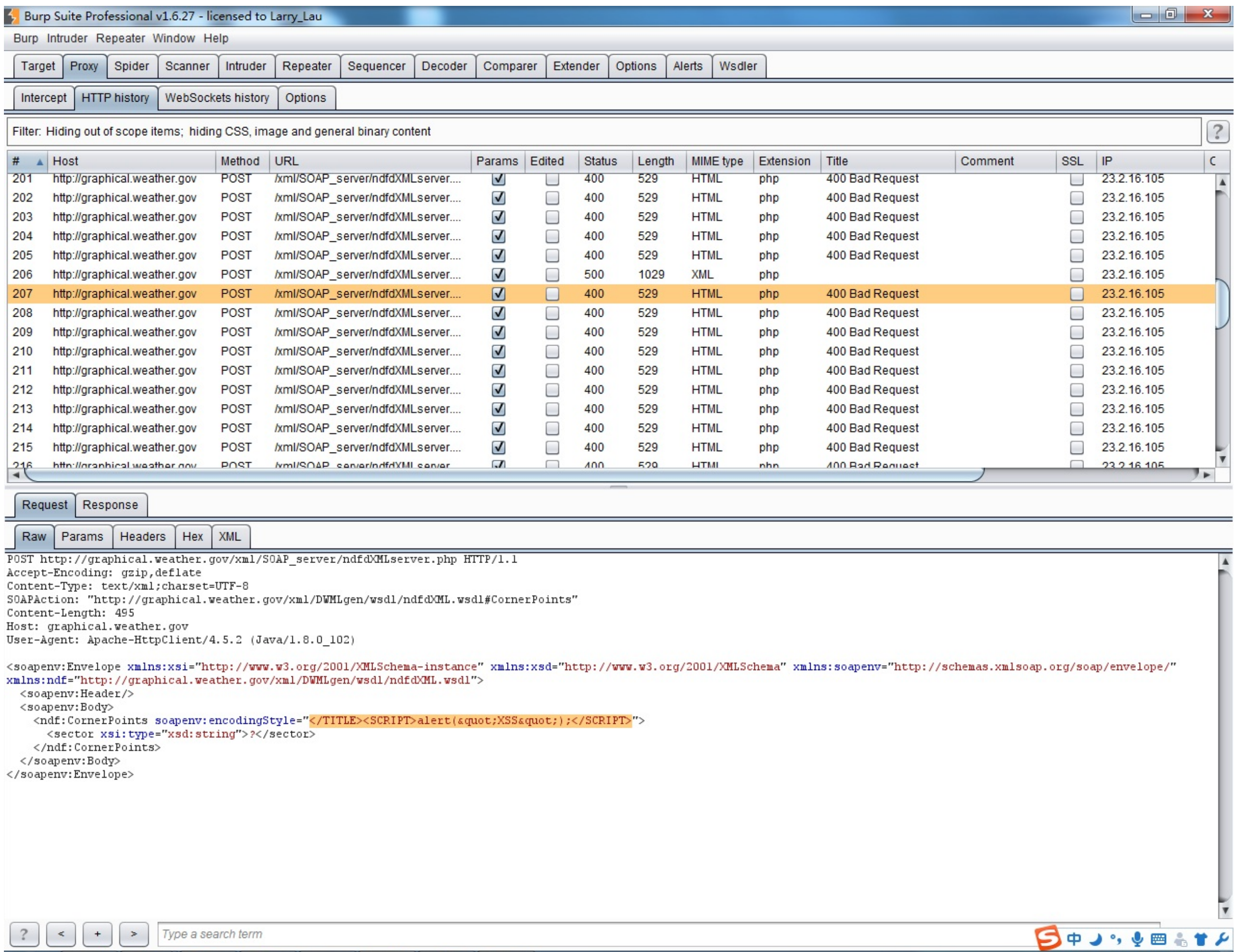
5.我们可以选择指定的SOAPAction或者某个SOAPAction下的某个安全项进行单一测试，也可以直接点击run运行所有的安全测试项。如果测试项过多的话，此操作执行时间会比较长，同时，如果并发数过多，会给服务器端造成压力，这是测试时候需要注意的。如下图所示，图中WebService接口正在安全测试中，进度条中显示调用的SOAPAction名称。



6.如果出现下图的状态，则表示测试进程已经执行完毕。



7.7. 此时，我们可以在Burp的Http history面板中查询到刚才发生的所有请求消息，通过不同的过滤条件查找我们关心的请求或响应消息，并发送到Burp的其他工具组件进行消息重放和处理、验证。

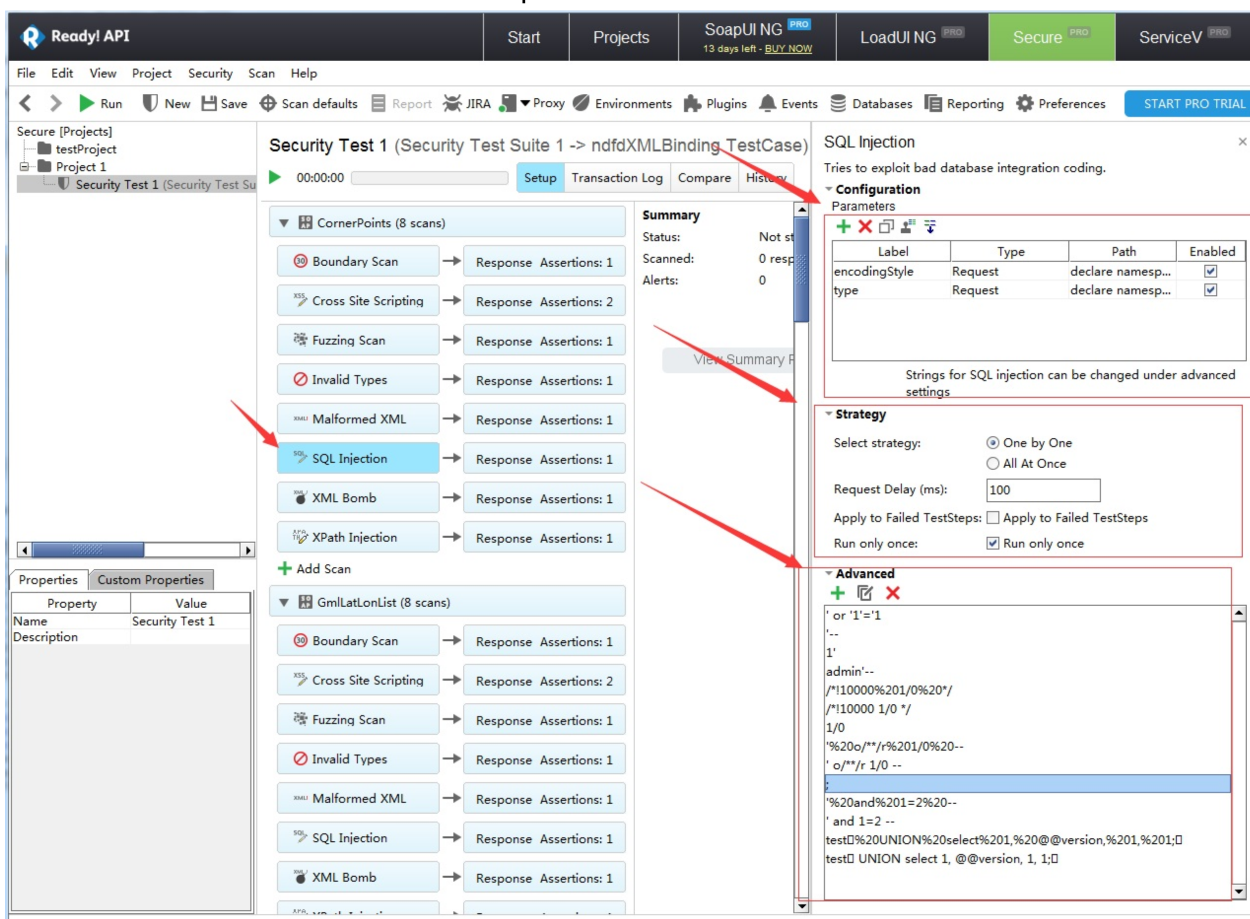


更多关于SOAPUI的使用请阅读这里

SoapUI NG Pro的安全测试项包括以下内容：

- 边界扫描
- SQL注入
- XPath/XQuery注入
- 模糊测试
- 无效的参数类型
- XML格式畸形
- XML炸弹
- 跨站脚本
- 上传附件安全
- 自定义扫描

下面就以SQL注入为例，我们看看SoapUI NG Pro的安全测试配置参数。



对于每一个安全测试项，其基本配置主要分三部分： 1.配置项（Configuration）

> 主要是指协议描述中定义的输入参数、编码类型、SOAP协议中的特定参数（namespace、import....)

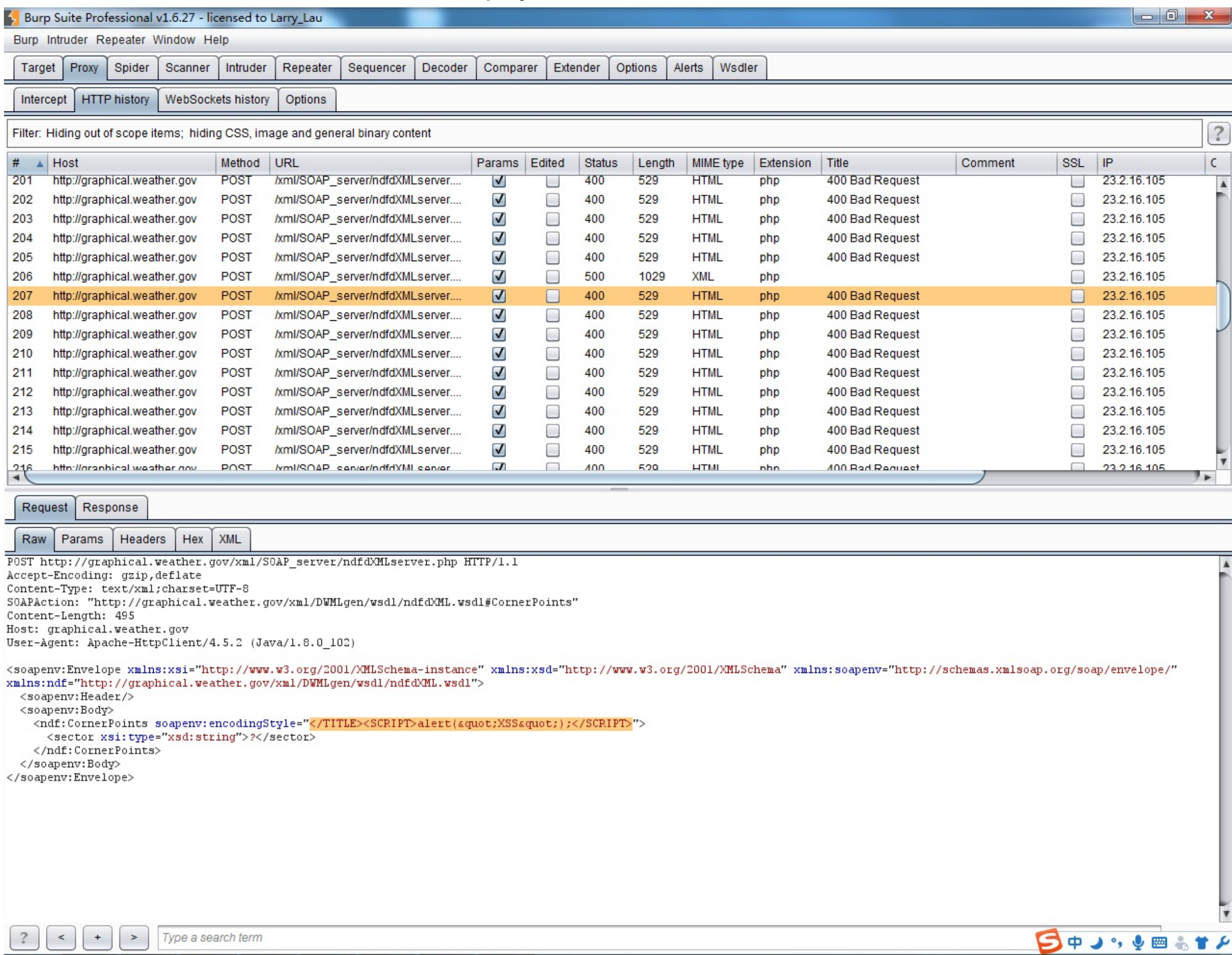
2.自动化测试策略（Straegy）

> 主要设置测试过程中的请求延时、选择策略、运行方式等

3.高级选项（Advanced）

> 通常是指测试时所需要的payload值，或者生成payload的策略。通过上图我们也可以看出，payload的值是可以自定义添加的。在github上，fuzzdb是被广泛使用的字典库，我们可以使用它作为测试的payload字典。项目地址为：https://github.com/fuzzdb-project/fuzzdb

当我们配置完毕后，运行安全测试项时，可以在Burp中查看到发送的payload值，如下图（阴影选中部分）所示的XSS脚本测试的payload：



同时，我们根据http状态码，对应答进行排序，跟踪可疑的响应消息，获取服务器的敏感信息。如下图获取的服务器Banner信息：

Burp Intruder Repeater Window Help

Target | Proxy | Spider | Scanner | Intruder | Repeater | Sequencer | Decoder | Comparer | Extender | Options | Alerts | Wsdler

Intercept | HTTP history | WebSockets history | Options

Filter: Hiding out of scope items; hiding CSS, image and general binary content

# ▲	Host	Method	URL	Params	Edited	Status	Length	MIME type	Extension	Title	Comment
175	http://graphical.weather.gov	POST	/xml/SOAP_server/ndfdXMLserver....	☑	☐	500	1029	XML	php		
176	http://graphical.weather.gov	POST	/xml/SOAP_server/ndfdXMLserver....	☑	☐	400	529	HTML	php	400 Bad Request	
177	http://graphical.weather.gov	POST	/xml/SOAP_server/ndfdXMLserver....	☑	☐	400	529	HTML	php	400 Bad Request	
178	http://graphical.weather.gov	POST	/xml/SOAP_server/ndfdXMLserver....	☑	☐	400	529	HTML	php	400 Bad Request	
179	http://graphical.weather.gov	POST	/xml/SOAP_server/ndfdXMLserver....	☑	☐	400	529	HTML	php	400 Bad Request	
180	http://graphical.weather.gov	POST	/xml/SOAP_server/ndfdXMLserver....	☑	☐	400	529	HTML	php	400 Bad Request	
181	http://graphical.weather.gov	POST	/xml/SOAP_server/ndfdXMLserver....	☑	☐	400	529	HTML	php	400 Bad Request	
182	http://graphical.weather.gov	POST	/xml/SOAP_server/ndfdXMLserver....	☑	☐	400	529	HTML	php	400 Bad Request	
183	http://graphical.weather.gov	POST	/xml/SOAP_server/ndfdXMLserver....	☑	☐	400	529	HTML	php	400 Bad Request	
184	http://graphical.weather.gov	POST	/xml/SOAP_server/ndfdXMLserver....	☑	☐	400	529	HTML	php	400 Bad Request	
185	http://graphical.weather.gov	POST	/xml/SOAP_server/ndfdXMLserver....	☑	☐	400	529	HTML	php	400 Bad Request	
186	http://graphical.weather.gov	POST	/xml/SOAP_server/ndfdXMLserver....	☑	☐	400	529	HTML	php	400 Bad Request	
187	http://graphical.weather.gov	POST	/xml/SOAP_server/ndfdXMLserver....	☑	☐	400	529	HTML	php	400 Bad Request	
188	http://graphical.weather.gov	POST	/xml/SOAP_server/ndfdXMLserver....	☑	☐	400	529	HTML	php	400 Bad Request	
189	http://graphical.weather.gov	POST	/xml/SOAP_server/ndfdXMLserver....	☑	☐	400	529	HTML	php	400 Bad Request	

Request | Response

Raw | Headers | Hex | HTML | Render

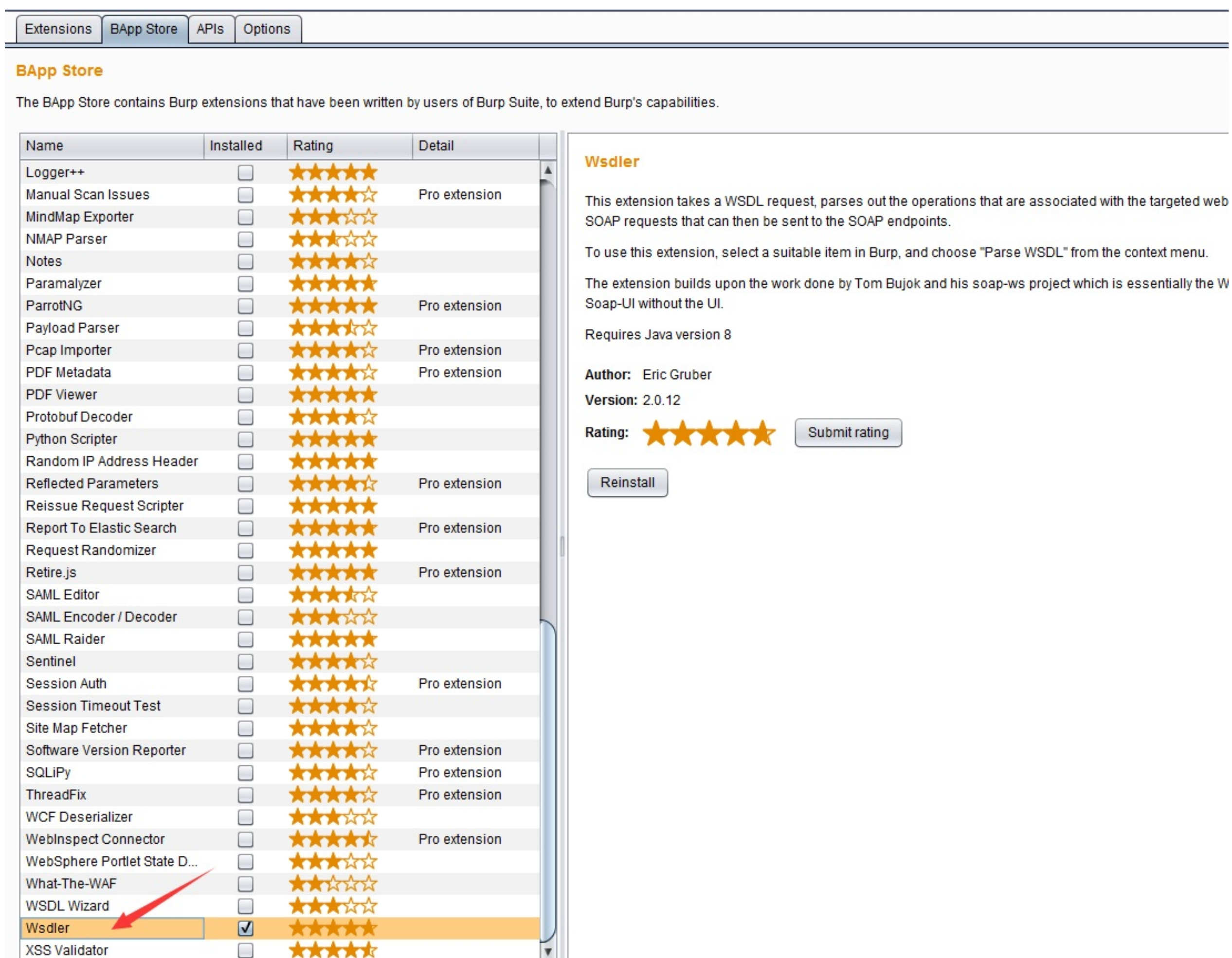
```
HTTP/1.1 400 Bad Request
Server: Apache/2.2.15 (Red Hat)
X-NIDS-ServerID: www5.mo
Content-Length: 314
Content-Type: text/html; charset=iso-8859-1
Date: Tue, 22 Nov 2016 06:55:18 GMT
Proxy-Connection: close

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not understand.<br />
</p>
<hr>
<address>Apache/2.2.15 (Red Hat) Server at graphical.weather.gov Port 80</address>
</body></html>
```

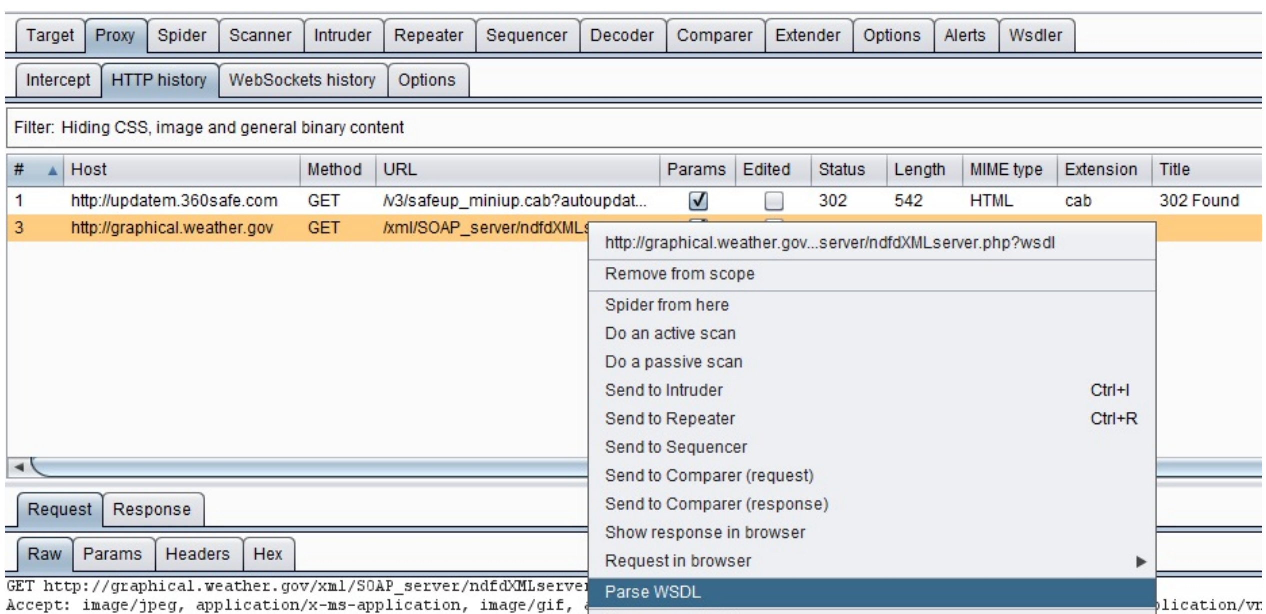
被Burp拦截到的消息记录，我们可以发送到Intruder，使用fuzzdb进行指定的fuzz测试；也可以发送到Repeater进行手工的消息内容修改和漏洞是否存在性的验证。具体到某个方面的漏洞，比如说Xpath注入漏洞，在测试过程中，需要测试人员理解Xpath的注入原理，理解Xpath的语法，根据服务器端的响应消息，自己手工构造特定的payload才能获得更重要的信息。这些都是在平时的工作中慢慢积累的，而不是光靠一款工具软件就作为万能的解决方案，希望读者能明白这个道理。

使用**Wsdler**测试**WebService**接口：

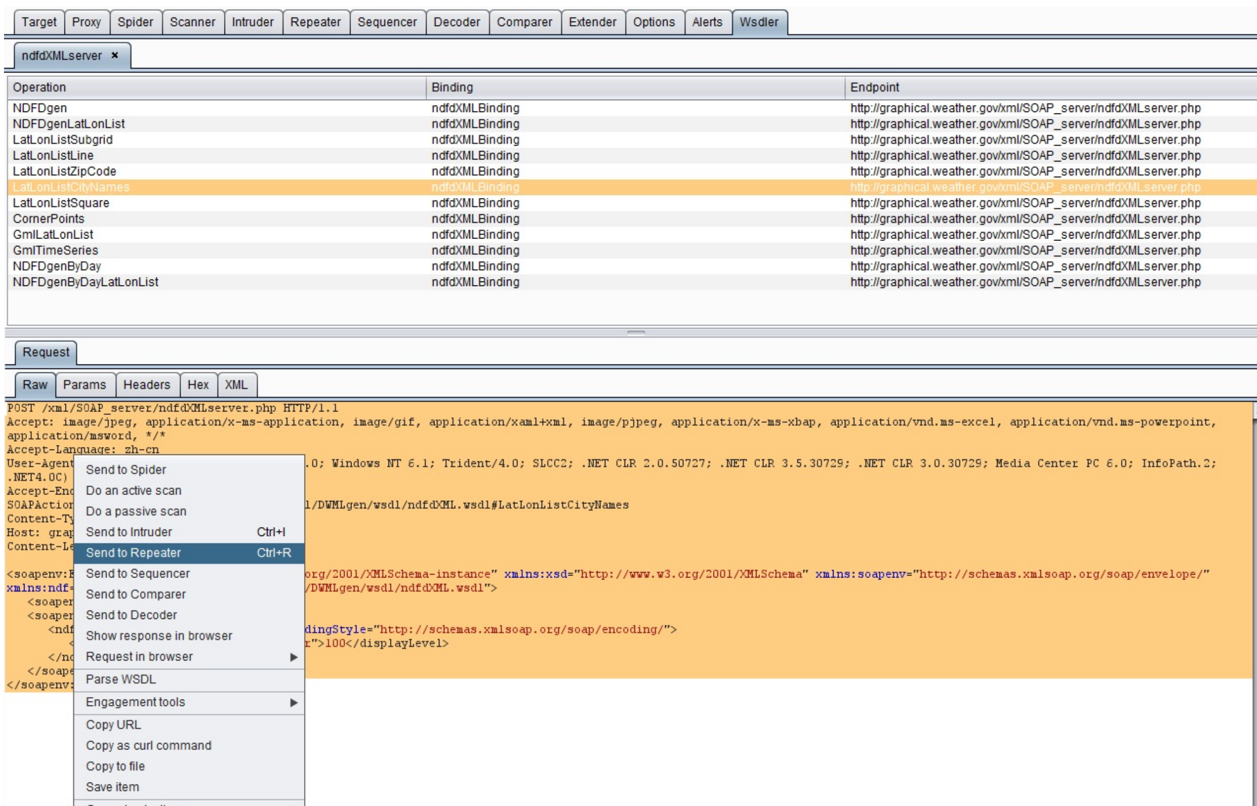
除了前面我们说的使用SOAPUI NG Pro 测试WebService外，在Burp里也有一个通过WSDL解析接口定义，手工测试WebService的插件：Wsdler



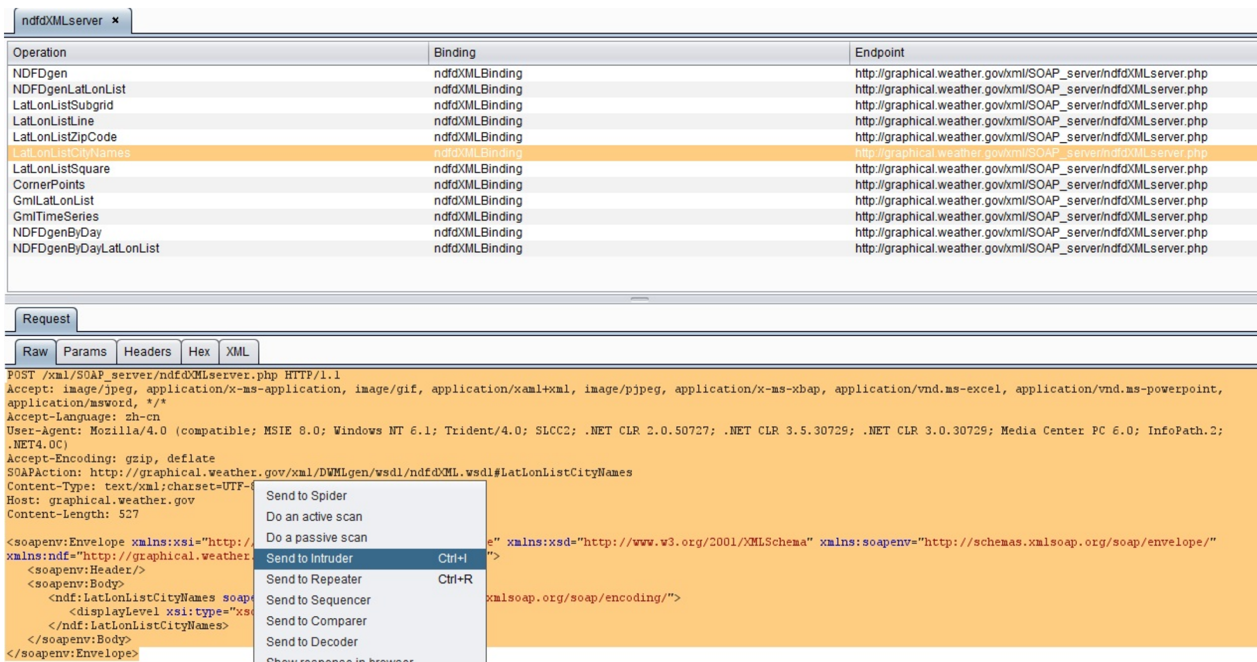
如果你安装了此插件，则在Burp的 Proxy >> History 中，可以直接使用【Parse WSDL】功能。



确认使用【Parse WSDL】解析功能后，此插件自动解析出服务的Operation、Binding、Endpoint。当选中某个Operation之后，可以查看SOAP消息文本。同时，可以发送到Burp的其他组件进行进一步操作。



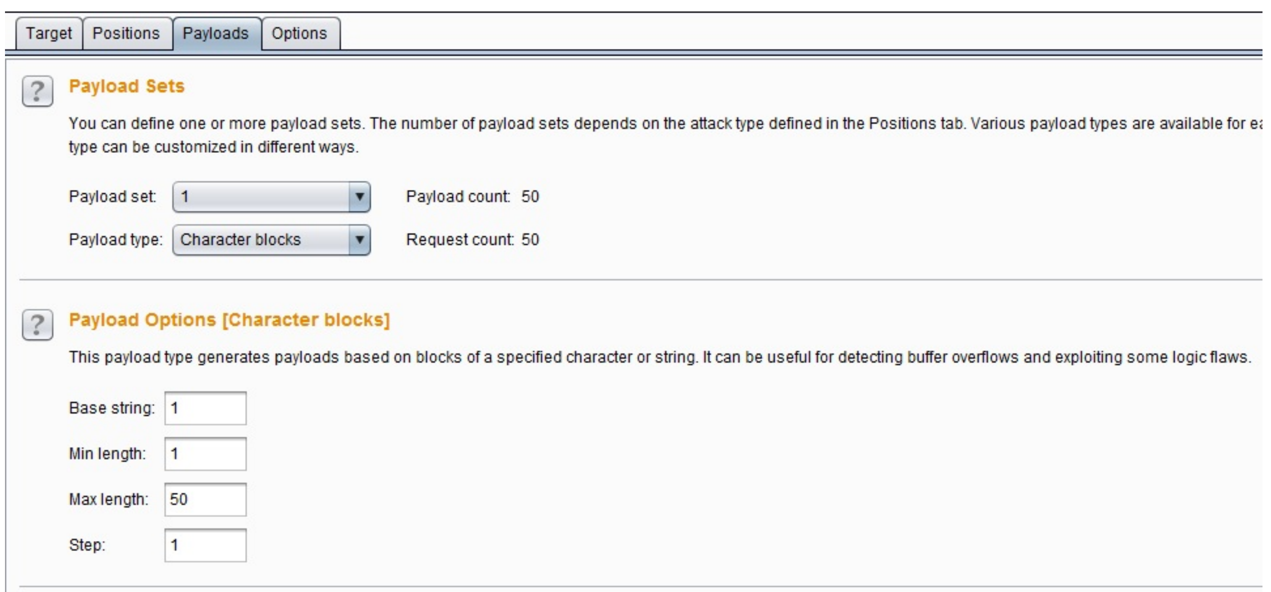
比如，我们将上图中的消息发送到Intruder，使用字符块（Character blocks）的对参数进行边界测试。



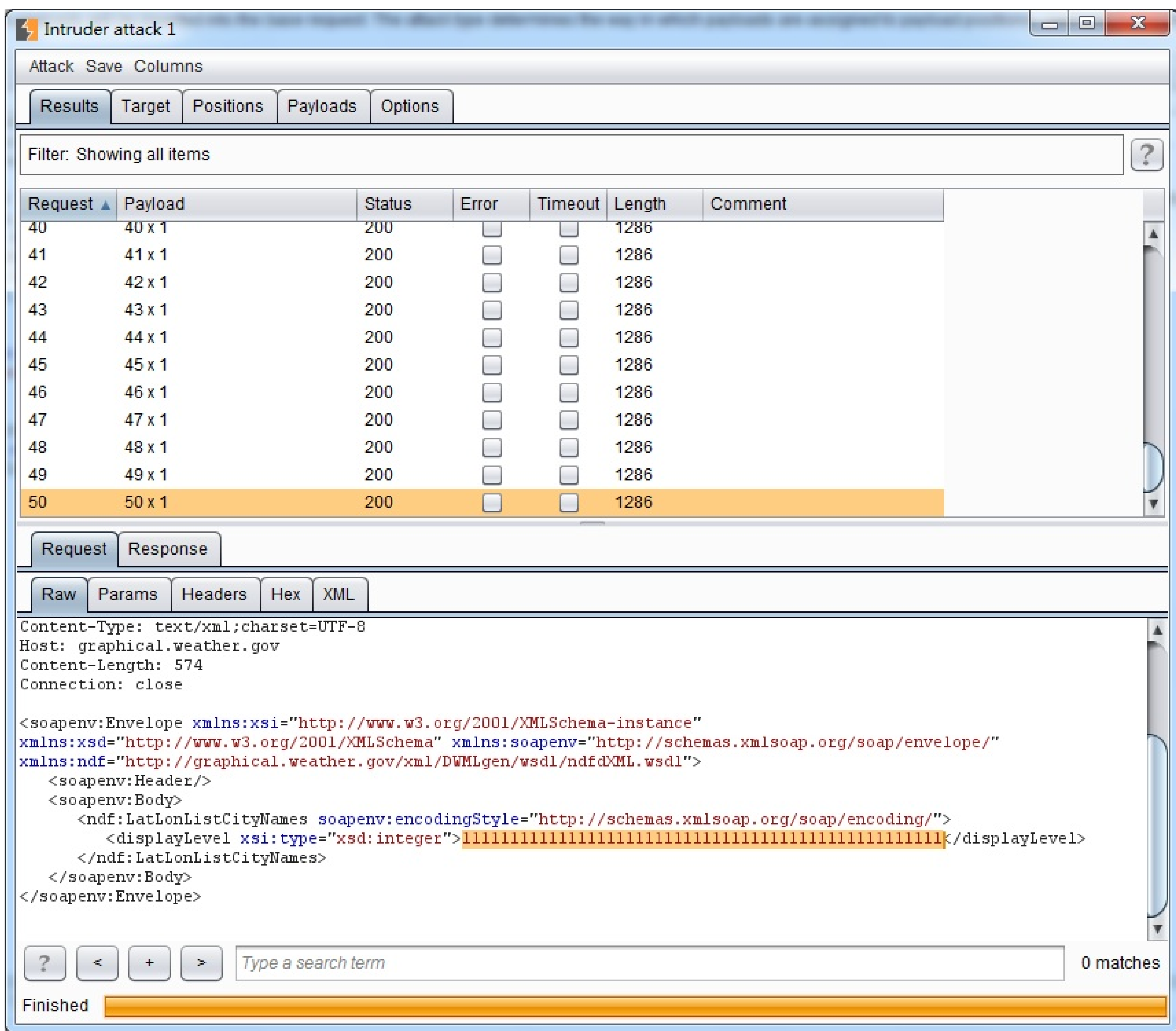
发送Intruder后的截图如下：



使用的payload为字符串1，从1到50，即1,11,111,1111......直到50个1，来测试参数的边界长度：



生成payload并执行后的结果如下图所示：



上面仅仅简单地叙述了Wsdler的使用，在实际的安全测试中，你可以使用Fuzzdb的字典，进行更复杂的渗透测试和功能验证。无论你使用什么样的工具，只要能通过一系列的自动化测试或者手工测试，完成WebService应用程序的安全脆弱性验证，保障应用程序的安全性，提供了应用程序的安全系统，这就达到我们做渗透测试的目的了。

使用Burp, Sqlmap进行自动化SQL注入渗透测试

在OWSAP Top 10中，注入型漏洞是排在第一位的，而在注入型漏洞中，SQL注入是远比命令行注入、Xpath注入、Ldap注入更常见。这就是本章要讲述的主要内容：在web应用程序的渗透测试中，如何使用Burp和Sqlmap的组合来进行SQL注入漏洞的测试。在讲述本章内容之前，默认为读者熟悉SQL的原理和SqlMap的基本使用，如果有不明白的同学，请先阅读《SQL注入攻击与防御》一书和SqlMap手册（最好是阅读官方文档）。

本章包含的内容有：

1. 使用gason插件+SqlMap测试SQL注入漏洞
2. 使用加强版sqlmap4burp插件+SqlMap批量测试SQL注入漏洞

使用gason插件+SqlMap测试SQL注入漏洞

在正式开始本章的内容之前，我们先做如下两点约定：

- 你已经安装配置好了python可运行环境
- 你已经熟悉sqlmap的基本命令行的使用并正确安装

如果你已经做到了上面的两点，那么，我们正式开始进入本章的内容。

Burp Suite与SqlMap整合的插件除了BApp Store 中的SQLiPy外（如图），

BApp Store

The BApp Store contains Burp extensions that have been written by users of Burp Suite, to extend Burp's capabilities.

Name	Installed	Rating	Detail
Logger++			
Manual Scan Issues			Pro extension
MindMap Exporter			
NMAP Parser			
Notes			
Paramalyzer			
ParrotNG			Pro extension
Payload Parser			
Pcap Importer			Pro extension
PDF Metadata			Pro extension
PDF Viewer			
Protobuf Decoder			
Python Scripter			
Random IP Address Header			
Reflected Parameters			Pro extension
Reissue Request Scripter			
Report To Elastic Search			Pro extension
Request Randomizer			
Retire.js			Pro extension
SAML Editor			
SAML Encoder / Decoder			
SAML Raider			
Sentinel			
Session Auth			Pro extension
Session Timeout Test			
Site Map Fetcher			
Software Version Reporter			Pro extension
SQLiPy			Pro extension
ThreadFix			Pro extension
WCF Deserializer			
WebInspect Connector			Pro extension
WebSphere Portlet State D...			
What-The-WAF			
WSDL Wizard			
Wsdler			
XSS Validator			

SQLiPy

This extension integrates Burp Suite with SQLMap.

Requirements:

- Jython 2.7 beta, due to the use of json.
- Java 1.7 or 1.8 (the beta version of Jython 2.7 requires this).
- A running instance of the SQLMap API server.

SQLMap comes with a RESTful based server that will execute SQLMap scans. You can manually start the server with:

```
python sqlmapapi.py -s -H <ip> -p <port>
```

Alternatively, you can use the SQLMap API tab to select the IP/Port on which to run, as well as the path to python and sqlmapapi.py on your system.

Once the SQLMap API is running, you just need to right-click in the 'Request' sub tab of either the Target or Proxy main tabs and choose 'SQLiPy Scan' from the context menu. This will populate the SQLMap Scanner tab with information about that request. Clicking the 'Start Scan' button will execute a scan. If the page is vulnerable to SQL injection, then these will be added to the Scanner Results tab.

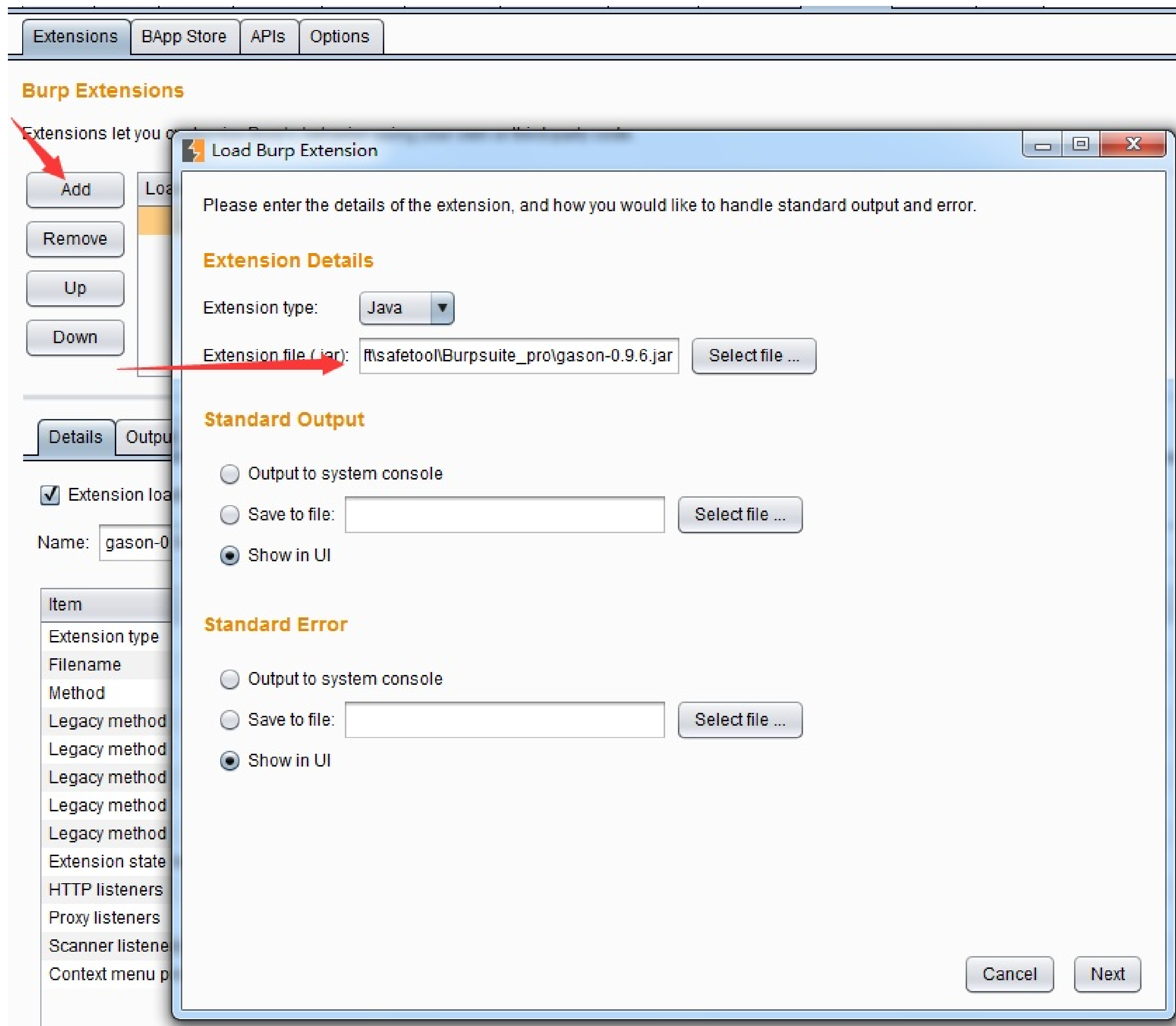
Author: Josh Berry @ CodeWatch
Version: 0.5.2
Rating: Submit rating

Install

还有gason和sqlmap4burp。不同的插件之间的功能大同小异，其目的都是使用命令行调用SqlMap的API接口进行SQL注入的测试，这里，我们主要以gason为例，讲述具体配置安装和功能使用。

gason插件安装使用大体分以下几个步骤：

1. 首先是下载gason插件。你可以从这个地址进行下载（点击下载），也可以从官方下载源码自己编译，总之就是获取到插件的安装文件gason-version.jar
2. 打开Burp Extensions进行安装，点击【Add】按钮，按照图中所示操作即可。安装过程很简单，如果不明白的话，可参考《Burp Suite应用商店插件的使用》章节的内容。



如果出现了下图中所示结果，且【Output】和【Errors】两个tab页面中没有错误的提示信息，表示插件已安装成功。

Burp Extensions

Extensions let you customize Burp's behavior using your own or third-party code.

Add | Remove | Up | Down

Loaded	Type	Name
☑	Java	gason-0.9.6.jar

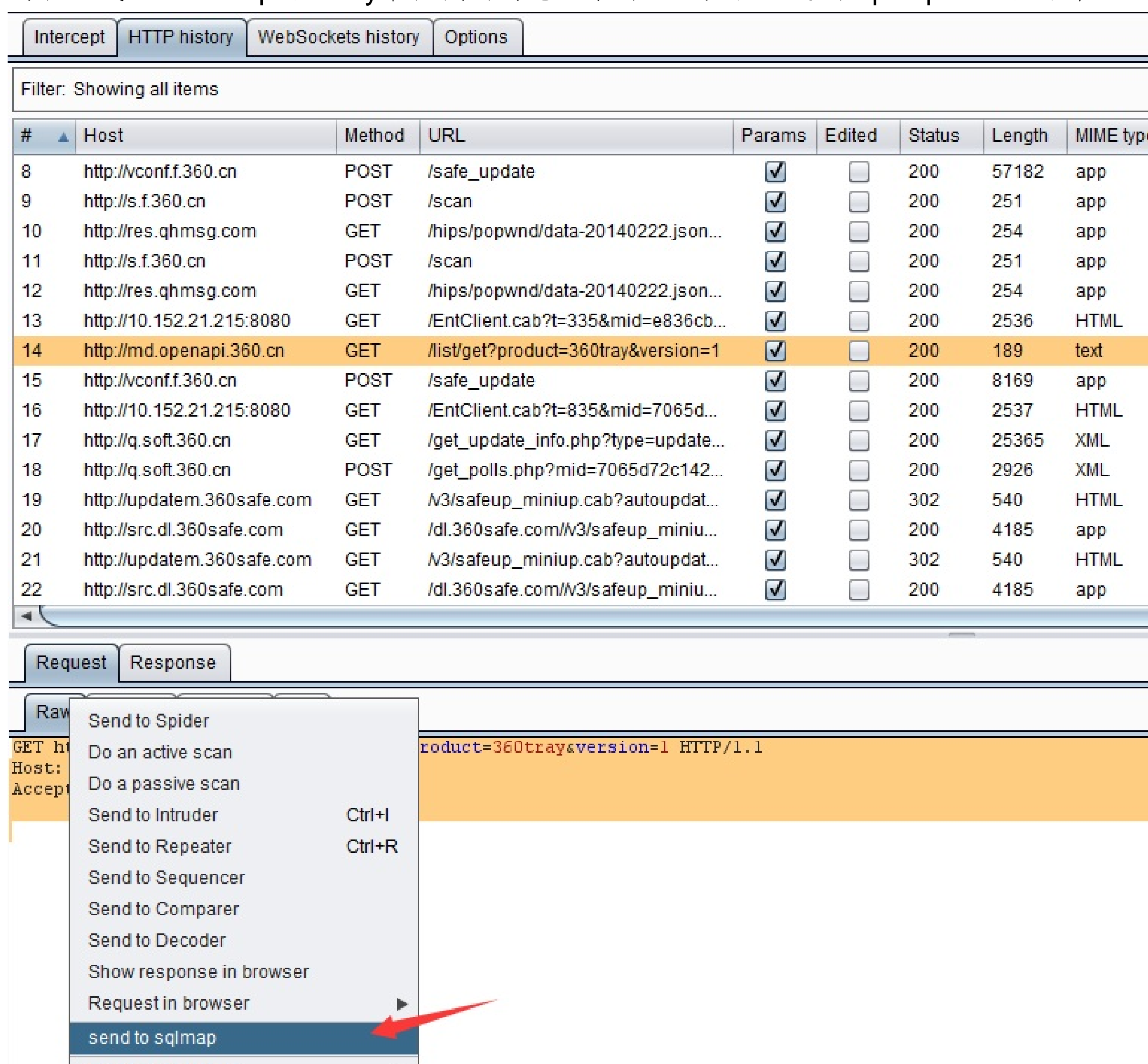
Details | Output | Errors

☑ Extension loaded

Name: gason-0.9.6.jar

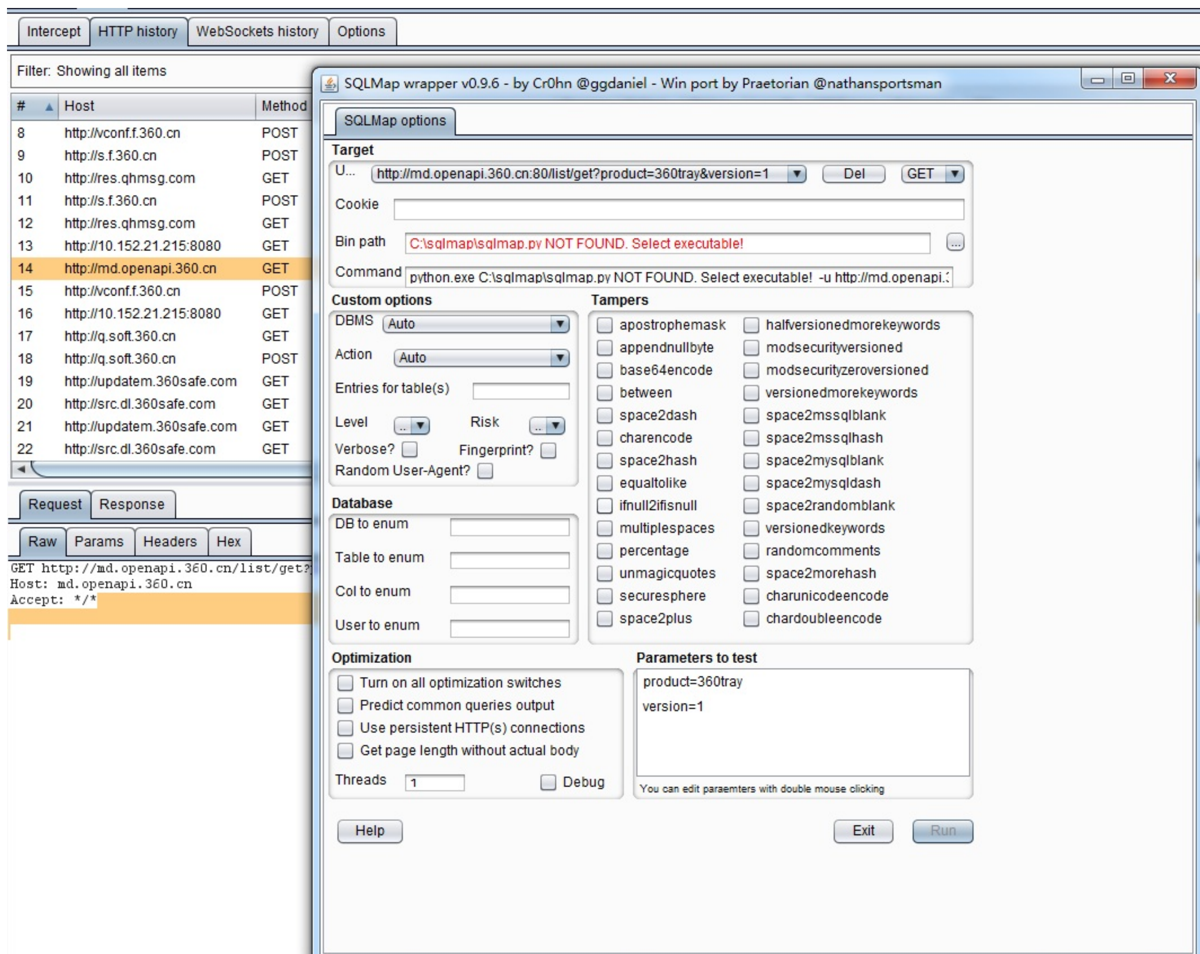
Item	Detail
Extension type	Java
Filename	D:\soft\safetool\Burpsuite_pro\gason-0.9.6.jar
Method	registerExtenderCallbacks
Legacy method	setCommandLineArgs
Legacy method	processHttpMessage
Legacy method	processProxyMessage
Legacy method	applicationClosing
Legacy method	newScanIssue
Extension state listeners	1
HTTP listeners	1
Proxy listeners	1
Scanner listeners	1
Context menu providers	1

3. 安装完成后，当Burp的Proxy中拦截到消息记录时，可直接发送到sqlmap。如下图所示：



4. 如果没有出现如上图所示的【send to sqlmap】菜单，则表示插件没正确安装成功，需要读者自己排查一下安装失败的原因。

5. 当我们在Burp拦截的请求消息上选择【send to sqlmap】后，则自动弹出sqlmap选项设置对话框。



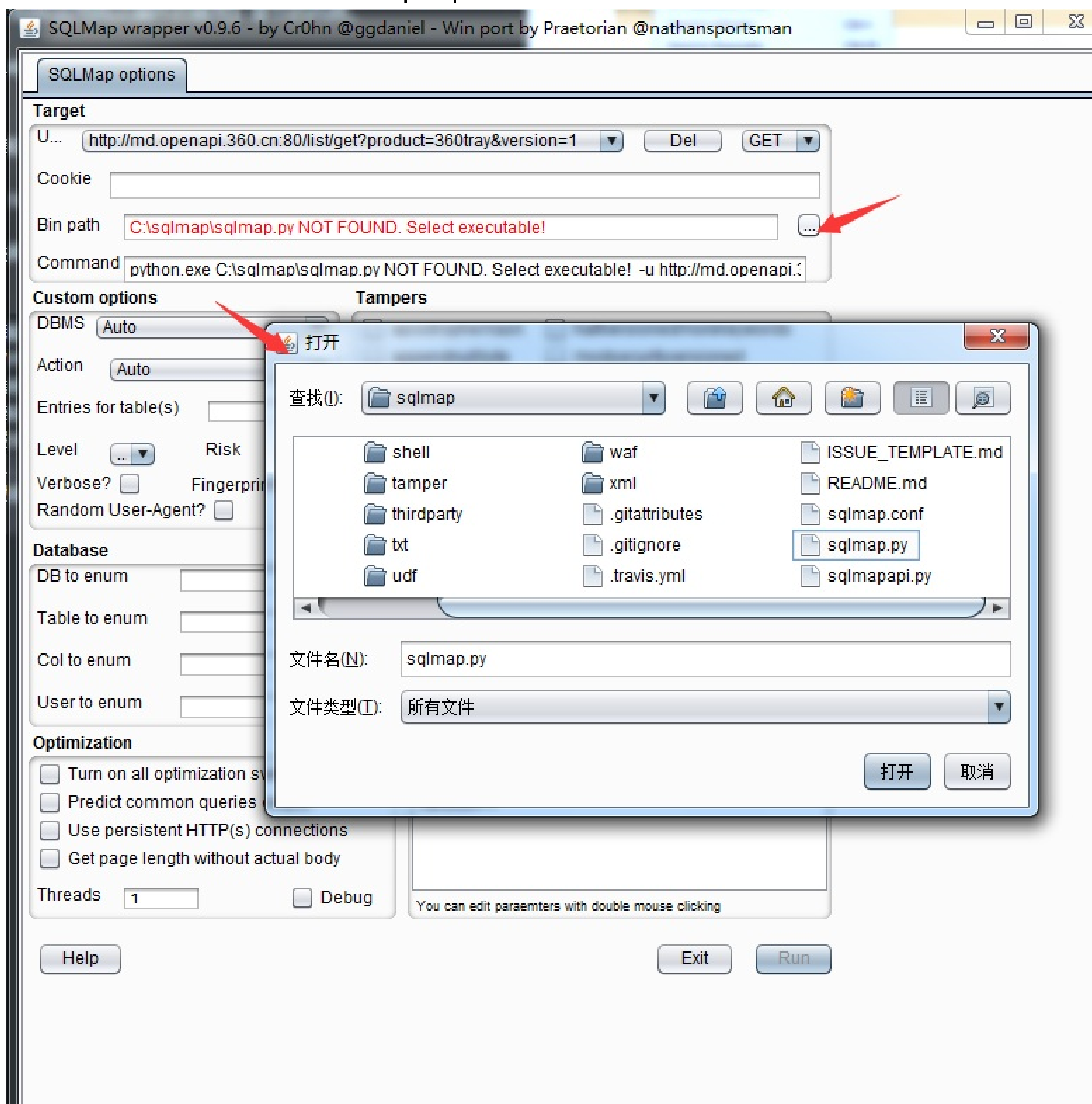
从图中我们可以看出，插件会自动抓取消息内容并解析后填充到相关参数设置的选项里去。例如：参数和参数值，请求方式（GET/POST），url地址等。同时，还有许多与Sqlmap本身测试使用的选项值仍需要我们自己指定，其中最主要的两个是：

bin目录：这里是指sqlmap.py的路径

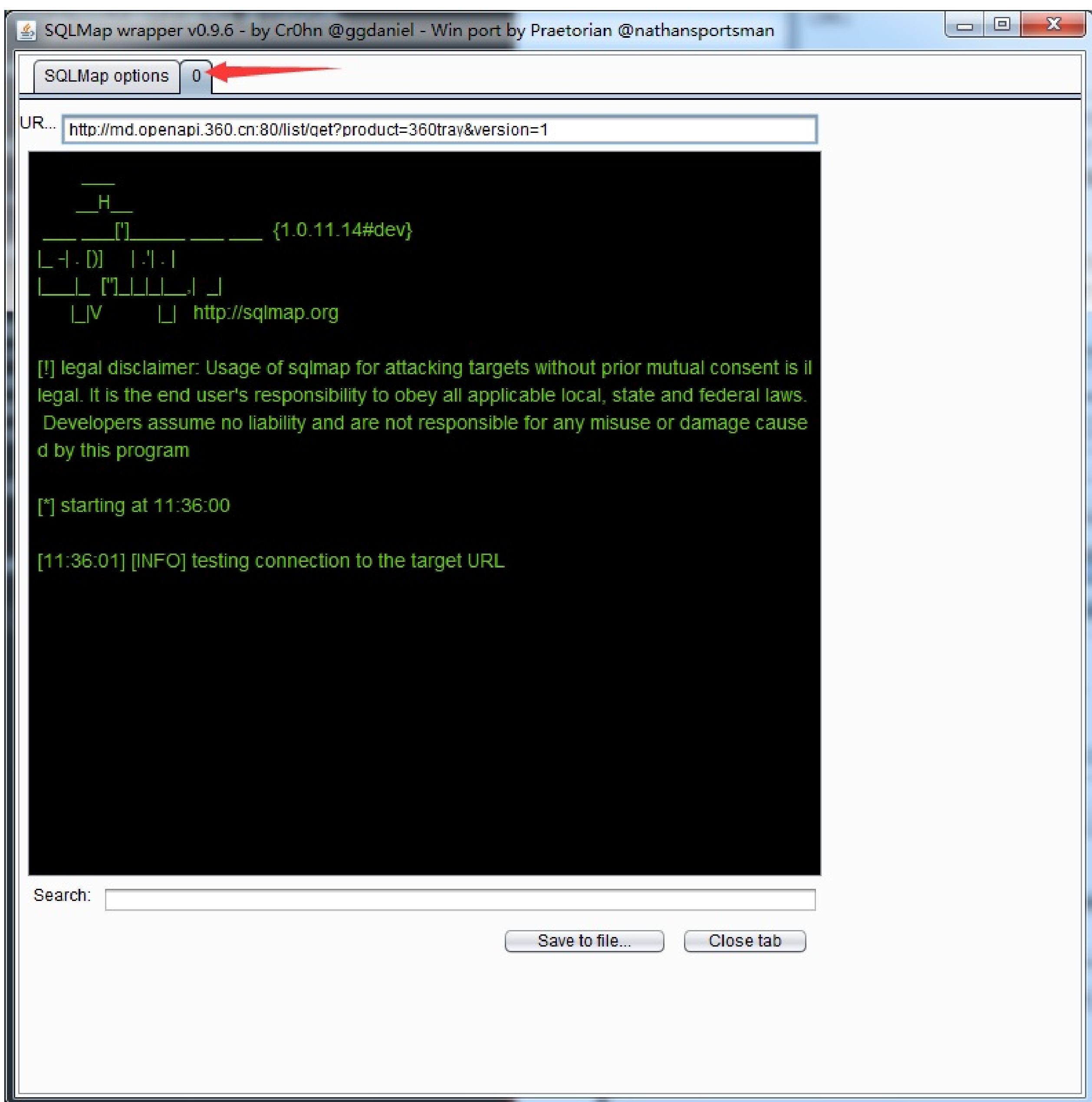
Command：sqlmap运行时执行的命令行

6.设置bin目录的方式很简单，点击【....】按钮，选择到sqlmap.py的存储路径即可。当bin path配置正确后，下方的Command会自动更新，随着设置参数的不同，自动调整需要执行的sqlmap命令行（如果不理解界面操作各个设置的含义，可以比较设置前后Command值的变化，即可以知道某个设置对应于sqlmap参数的哪一个选项）。

6.设置bin目录的方式很简单，点击【....】按钮，选择到sqlmap.py的存储路径即可。当bin path配置正确后，下方的Command会自动更新，随着设置参数的不同，自动调整需要执行的sqlmap命令行（如果不理解界面操作各个设置的含义，可以比较设置前后Command值的变化，即可以知道某个设置对应于sqlmap参数的哪一个选项）。



7.所有的配置正确之后，【run】按钮将被激活，点击【run】，系统自动进入sqlmap扫描阶段。



当进入sqlmap扫描阶段时，插件会新增一个tab页面，显示执行进度，即如上图的箭头所指。

8.我们可以通过进度跟踪的界面上的【save to file】和【close tab】来保存扫描结果和关闭、终止扫描。

使用gason插件，与命令行方式执行sqlmap脚本相比，操作变得更加方便。比如说，在命令行环境中，我们需要先抓取cookie信息，才能放入到命令行里执行；亦或者，我们需要手工录入一个个参数进行命令行操作，而在gason插件环境中，这些都不需要。当我们点击【send to sqlmap】时，插件自动帮我们完成了这些操作。且与sqlmap个性设置的选项，我们也可以通过界面操作，自动完成，比命令行下更直观、更高效。

使用加强版**sqlmap4burp**插件**+SqlMap**批量测试**SQL**注入漏洞

如果你只想执行一次sqlmap的操作，即能完成多个链接地址的SQL注入漏洞测试，使用gason插件的方式操作起来会比较麻烦。那么，是否存在批量检测的使用方法呢？国内比较著名的安全网站freebuf上有两篇类似的文章，感兴趣的同学可以自己阅读看看。

1. 【优化SQLMAP的批量测试能】http://www.freebuf.com/sectool/75296.html
2. 【我是如何打造一款自动化SQL注入工具】http://www.freebuf.com/sectool/74445.html

通过上面的两篇文章，我们可以看出，批量操作在实际应用中非常常见，如果能解决批量问题，则大大地提高了我们的工作效率，下面我们一起来研究一下如何解决这个问题。

在Sqlmap的官方文档中有这样的介绍：

sqlmap user's manual 　　　　　　　　　　　　　　　　*5 History*

5.13.3 Parse targets from Burp or WebScarab proxy logs

Option: `-l`

Rather than providing a single target URL, it is possible to test and inject against HTTP requests proxied through Burp proxy or WebScarab proxy. This option requires an argument which is the proxy's HTTP requests log file.

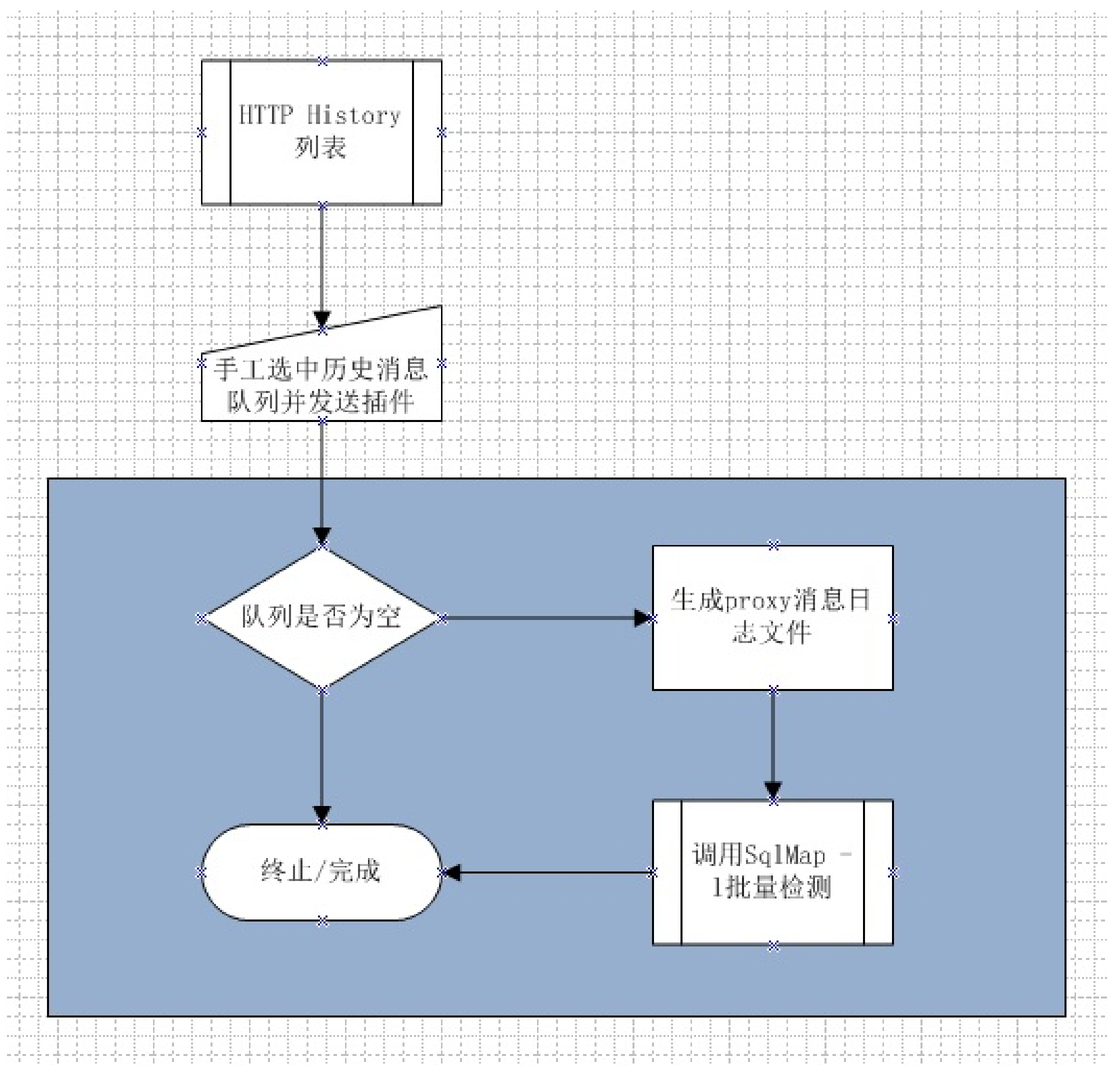
从这段话中我们可以看出，sqlmap可以通过 -l 参数，一次检测多个url的注入问题，这个参数的值是Burp proxy或者WebScarab proxy的日志文件。那么，我们是否可以通过插件的方式，自动生成类似的日志文件，然后调用sqlmap，解决批量检测的问题？答案当然也是肯定的。

在github上，网友difcareer公开了一个Burp插件sqlmap4burp，源文件地址为：https://github.com/difcareer/sqlmap4burp。我们就基于此插件的功能拓展，来完成自动化批量SQL测试的功能。

首先，我们来规划一下这个插件的使用场景：

> 当通过Burp代理的HTTP流量消息都记录在HTTP History 列表中，我们可以批量地选中多个url，由插件自动生成类似Burp proxy的日志文件，然后调用sqlmap进行检测。

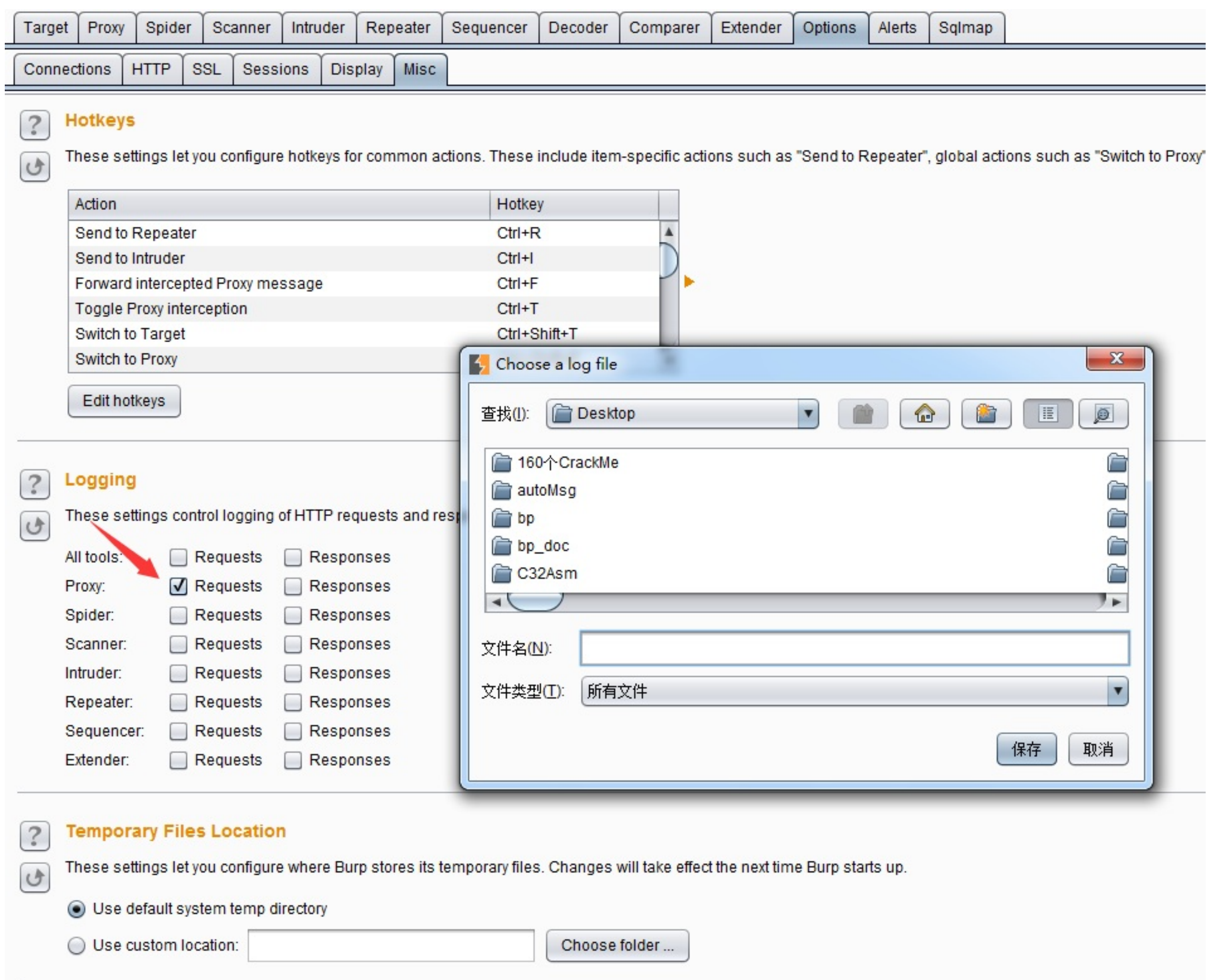
插件整个使用过程的流程图如下：



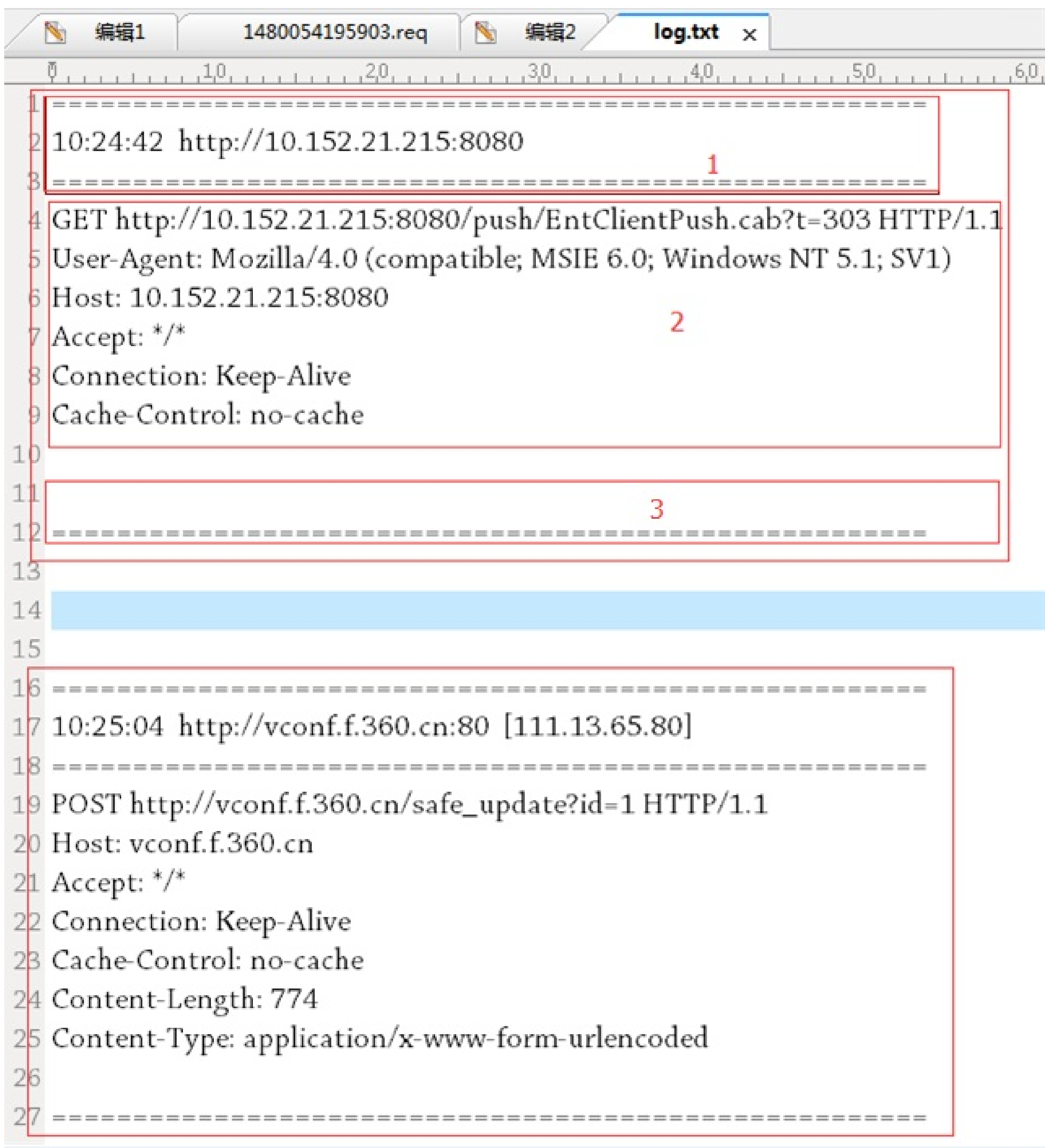
上图中浅蓝色背景标示的部分，均为插件所执行的动作。其主要做了这些事情：

1. 判断选中数据是否为空，不为空则获取History列表的已选中数据，无论一条还是多条记录。
2. 将获取的HTTP消息按照proxy日志的格式，生成日志文件。
3. 调用sqlmap.py脚本，传递生成的日志文件作为参数值进行检测。

明白了这些，接着我们来看proxy的日志文件格式。



如上图所示，我们通过【Options】>>【Misc】>>【Logging】选中Proxy的Requests选项，自动弹出保存日志文件的路径和文件名，点击【保存】按钮后，则文件生成并开始记录Proxy的请求消息。 我们把生成的日志文件用记事本打开后发现，日志格式如下：



上图一共两条消息，每一条消息内容又包含图中1的头部，图中2的消息内容和图中3的尾部构成，而图中2的部分即是消息请求的详细内容，则我们按照此格式手工构造日志文件，通过修改sqlmap4burp的源码（Windows环境下）从而来完成这个功能。

在源码SnifferContextMenuFactory.java的我们找到了日志获取的入口createMenuItems函数内部的actionPerformed函数，遂修改此段代码为：

```
@Override
public List<JMenuItem> createMenuItems(final IContextMenuInvocation invocation) {
    List<JMenuItem> list = new ArrayList<JMenuItem>();
    JMenuItem jMenuItem = new JMenuItem("send to Sqlmap");
    list.add(jMenuItem);
    jMenuItem.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            IHttpRequestResponse[] messages = invocation.getSelectedMessages();
            File file = new File(Context.getTempReqName(true));
            //循环遍历选中的消息，以append方式追加到日志文件中
            for(int i=0;i<messages.length;i++){
                byte[] req = messages[i].getRequest();
                try {
                    //添加单条的日志头部信息
                    FileUtils.writeByteArrayToFile(file,createLogHeader(messages[i]),true);
                    //添加单条的http信息
                    FileUtils.writeByteArrayToFile(file,req,true);
                  //添加单条的日志尾部信息
                    FileUtils.writeByteArrayToFile(file,createLogFooter(),true);
                } catch (IOException e1) {
                    e1.printStackTrace();
                }
            }
            System.out.println("sent to sqlMap");
            new Thread(new SqlmapStarter()).start();

        }
    });
    return list;
}
```

而创建日志头部和尾部的代码主要是拼写同格式的字符串，详细如下：

```
/**
 * 构造log日志头部信息，格式如：
 * =====================================================
 *  10:24:42  http://10.152.21.215:8080
 * =====================================================
 */
private byte[] createLogHeader(IHttpRequestResponse messages){
    StringBuffer sb = new StringBuffer();
    IRequestInfo analyzeRequest = helpers.analyzeRequest(messages); // 对消息体进行解析
    URL url = analyzeRequest.getUrl();
    sb.append("=====================================================\n");
    sb.append(getNowDate()+"  "+url.getProtocol()+"://"+url.getHost()+":"+url.getPort()+"\n");
    sb.append("=====================================================\n");
    return sb.toString().getBytes();
}

/**
 * 构造log日志尾部信息,格式如下：
 * =====================================================
 */
private byte[] createLogFooter(){
    StringBuffer sb = new StringBuffer();
    sb.append("=====================================================\n\n\n\n");
    return sb.toString().getBytes();
}
```

同时，修改sqlmap参数的调用方式，修改SqlmapStarter.java的第21行为：

```java
public class SqlmapStarter implements Runnable {

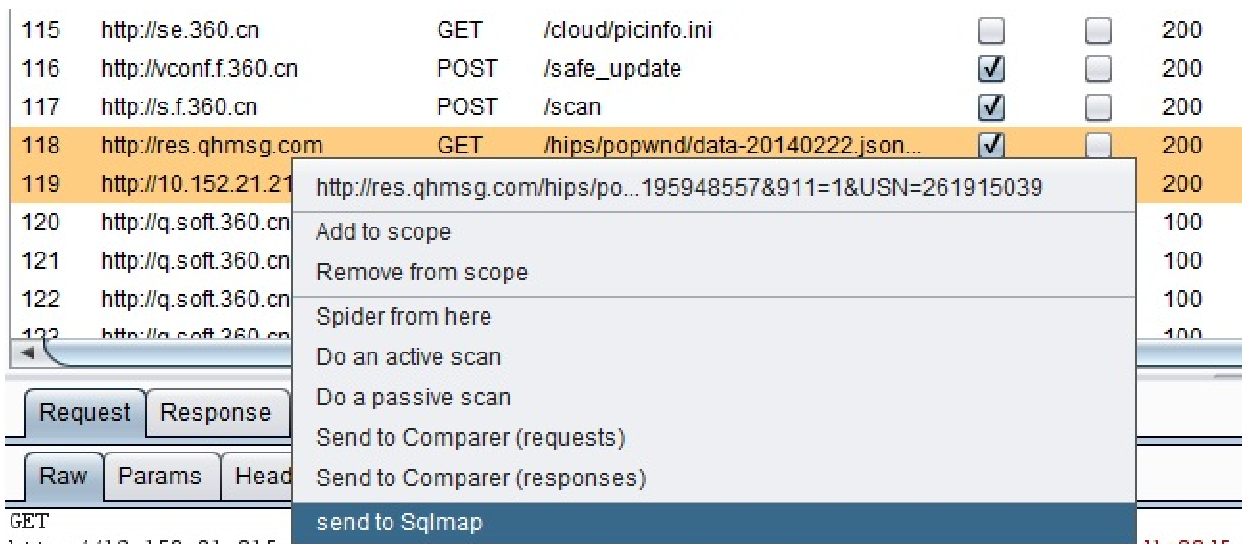

    @Override
    public void run() {
        try {
            StringBuilder sb = new StringBuilder();
            sb.append("sqlmap.py -l " + Context.getTempReqName(false)+" --batch -smart");
            if (isNotBlank(Context.userConfig)) {
                sb.append(" " + Context.userConfig);
            }
            File batFile = new File(Context.getTempBatName(true));
            if (!batFile.exists()) {
                batFile.createNewFile();
            }
```

这样，我们可以实现批量操作的功能了。

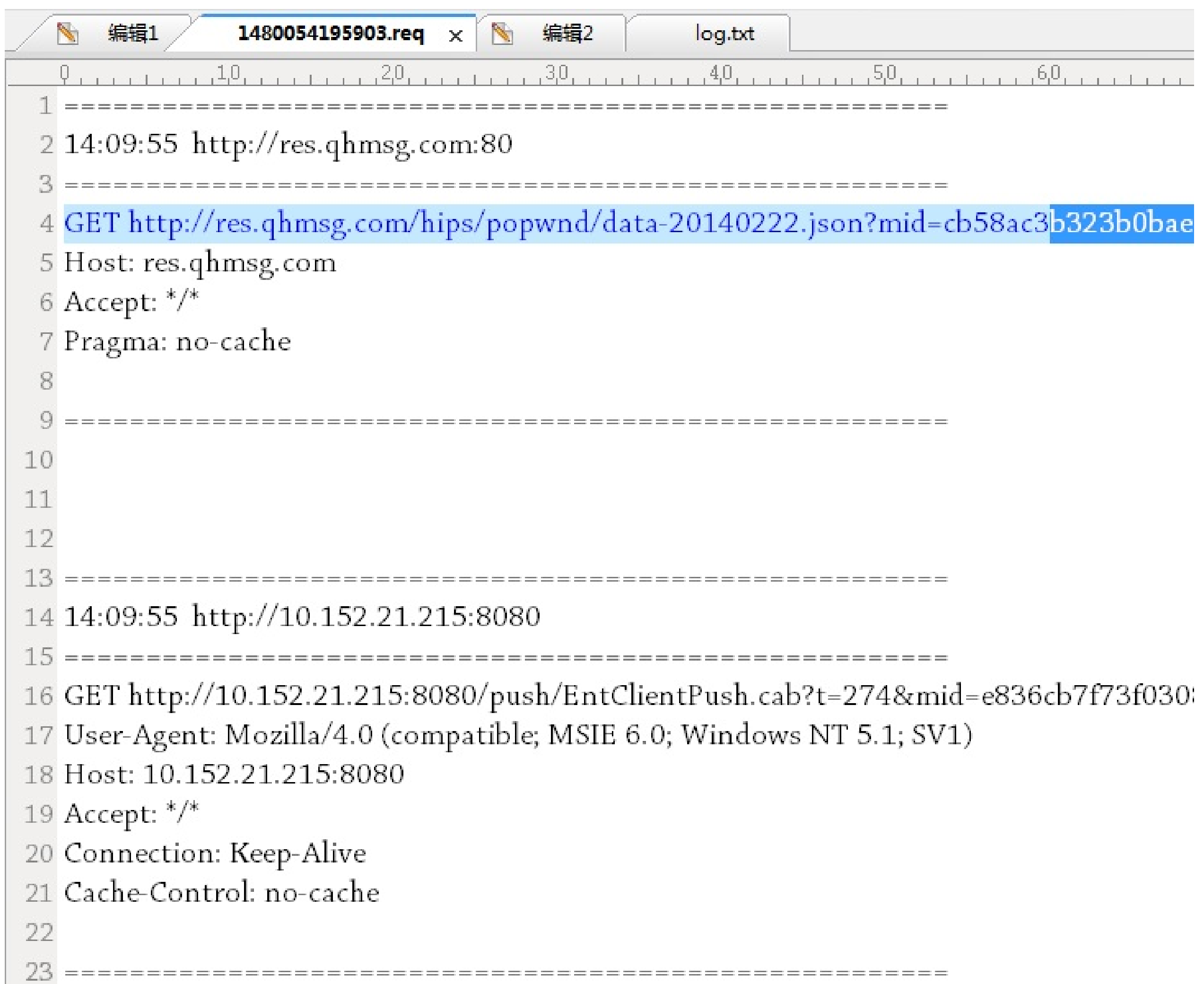
插件和源码可以通过如下地址进行下载：插件下载 源码下载

下载完毕后，请参考sqlmap4burp的readme完成基本的配置放可以使用，否则sqlmap调用将会失败，无法完成批量检测。

插件安装完毕后显示跟原来的插件并无多大区别，如下图是发送多条url到SqlMap的截图：



生成的日志文件的截图：



```
编辑1    1480054195903.req    编辑2    log.txt

1  ====================================================
2  14:09:55  http://res.qhmsg.com:80
3  ====================================================
4  GET http://res.qhmsg.com/hips/popwnd/data-20140222.json?mid=cb58ac3b323b0bae
5  Host: res.qhmsg.com
6  Accept: */*
7  Pragma: no-cache
8
9  ====================================================
10
11
12
13 ====================================================
14 14:09:55  http://10.152.21.215:8080
15 ====================================================
16 GET http://10.152.21.215:8080/push/EntClientPush.cab?t=274&mid=e836cb7f73f030
17 User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
18 Host: 10.152.21.215:8080
19 Accept: */*
20 Connection: Keep-Alive
21 Cache-Control: no-cache
22
23 ====================================================
```

sqlmap窗口中一次可以检测多个ur截图：

```
C:\Windows\system32\cmd.exe
C:\                sqlmap.py -l C:\Users\     AppData\Local\Temp\1480054195903
.req --batch -smart
        ___
       __H__
 ___ ___[,]_____ ___ ___  {1.0.11.14#dev}
|_ -| . [,]     | .'| . |
|___|_  [(]_|_|_|__,|  _|
      |_|V          |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent
 is illegal. It is the end user's responsibility to obey all applicable local, state and
 federal laws. Developers assume no liability and are not responsible for any misuse or
damage caused by this program

[*] starting at 14:14:46

[14:14:46] [INFO] sqlmap parsed 2 (parameter unique) requests from the targets list read
y to be tested
[14:14:46] [INFO] sqlmap got a total of 2 targets
URL 1:
GET http://res.qhmsg.com/hips/popwnd/data-20140222.json?mid=cb5              76cb24a
b92a&status=11&v=28        4718&r=195       911=1&USN=26
do you want to test this URL? [Y/n/q]
> Y
[14:14:46] [INFO] testing URL 'http://res.qhmsg.com/hips/popwnd/data-20140222.json?mid=c
b5              76cb24ab92a&status=11&v=281        18&r=19       a911=1&USN=26
039'
[14:14:46] [INFO] using 'C:\Users\AMCC\.sqlmap\output\results-11252016_0214pm.csv' as th
e CSV results file in multiple targets mode
[14:14:46] [INFO] testing connection to the target URL
[14:15:07] [CRITICAL] unable to connect to the target URL. sqlmap is going to retry the
request(s)
[14:15:07] [WARNING] if the problem persists please check that the provided target URL i
s valid. In case that it is, you can try to rerun with the switch '--random-agent' turne
d on and/or proxy switches ('--ignore-proxy', '--proxy',...)
[14:16:10] [ERROR] unable to connect to the target URL, skipping to the next URL
URL 2:
GET http://10.152.21.215:8080/push/EntClientPush.cab?t=274&mid=e83              54b91
d1e22d5&mac=c80aa9cdd88b&ip=1         7&host=AMCC-OP-3N052&sl=2016.11.25%2000:24:55.1&
ver=3.0.0.1546&dm=
```

使用Burp、PhantomJS进行XSS检测

XSS（跨站脚本攻击）漏洞是Web应用程序中最常见的漏洞之一，它指的是恶意攻击者往Web页面里插入恶意html代码，当用户浏览该页之时，嵌入其中Web里面的html代码会被执行，从而达到恶意攻击用户的特殊目的，比如获取用户的cookie，导航到恶意网站，携带木马等。根据其触发方式的不同，通常分为反射型**XSS**、存储型**XSS**和**DOM-base**型**XSS**。漏洞“注入理论”认为，所有的可输入参数，都是不可信任的。大多数情况下我们说的不可信任的数据是指来源于HTTP客户端请求的URL参数、form表单、Headers以及Cookies等，但是，与HTTP客户端请求相对应的，来源于数据库、WebServices、其他的应用接口数据也同样是不可信的。根据请求参数和响应消息的不同，在XSS检测中使用最多的就是动态检测技术：以编程的方式，分析响应报文，模拟页面点击、鼠标滚动、DOM 处理、CSS 选择器等操作，来验证是否存在XSS漏洞。

本章包含的内容有：

1. XSS漏洞的基本原理
2. PhantomJS在XSS检测中的使用原理
3. 使用XSS Validator插件进行XSS漏洞检测

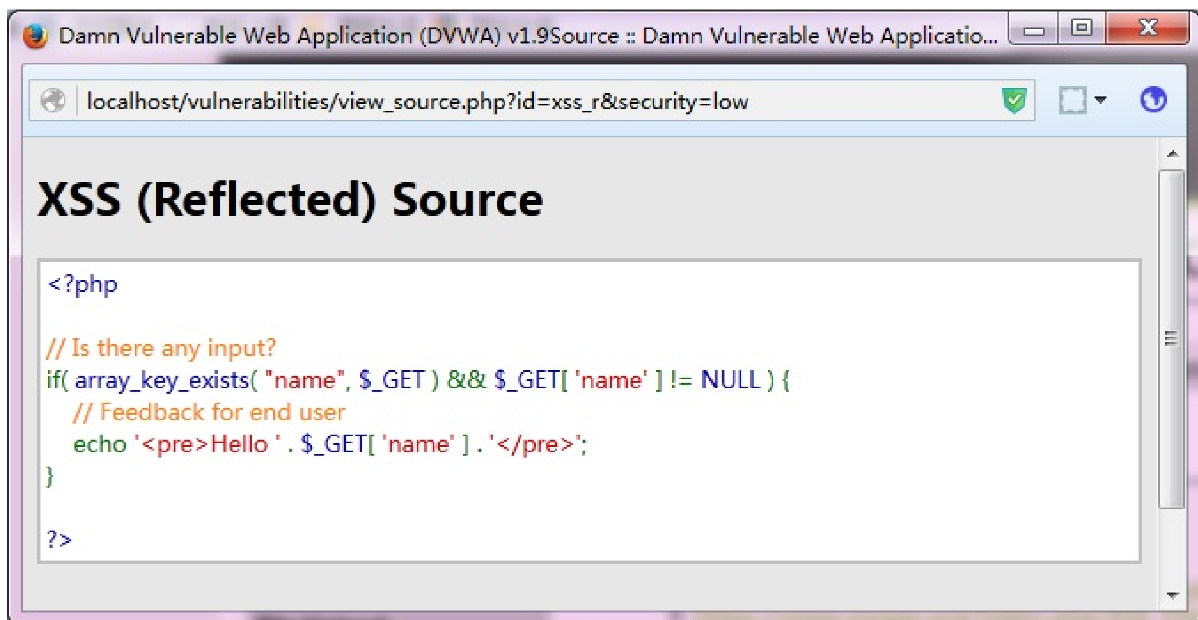
XSS漏洞的基本原理

一般来说，我们可以通过XSS漏洞的表现形式来区分漏洞是反射型、存储型、DOM-base三种中的哪一种类型。

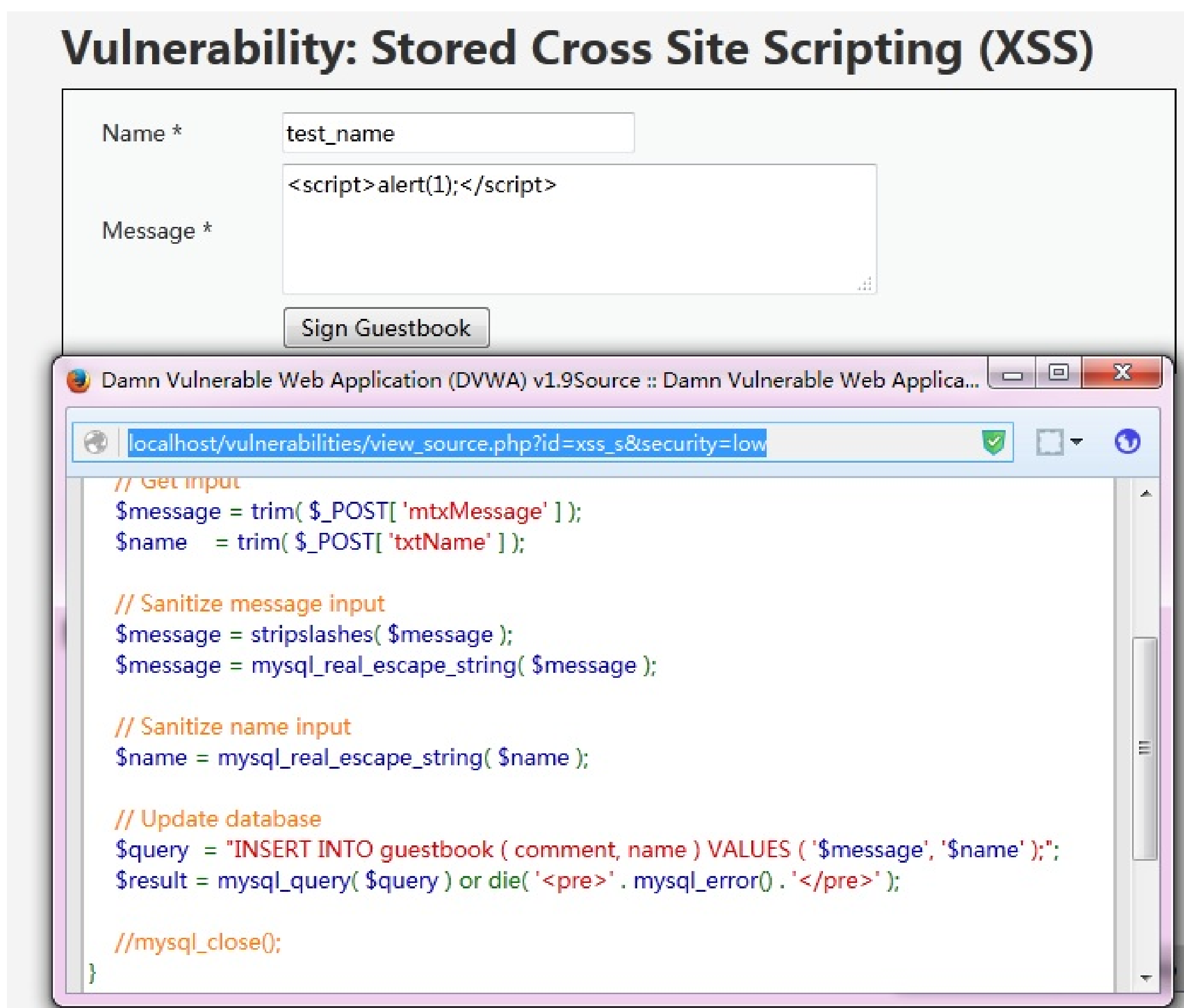
1. 反射型**XSS**是指通过给别人发送带有恶意脚本代码参数的URL，当URL地址被打开时，带有恶意代码参数被HTML解析、执行。它的特点是非持久化，必须用户点击带有特定参数的链接才能引起。它的连接形式通常如下：

   ```
   http://localhost/vulnerabilities/xss_r/?name=<script>alert(1);</script>
   ```

   其name参数的值为 `<script>alert(1);</script>` ，这样的参数值进入程序代码后未做任何处理，从而被执行。其类似的源代码如下图：



2. 存储型**XSS**是指恶意脚本代码被存储进数据库，当其他用户正常浏览网页时，站点从数据库中读取了非法用户存储的非法数据，导致恶意脚本代码被执行。通常代码结构如下图：



其发生XSS的根本原因是服务器端对写入数据库中的内容未做javascript脚本过滤。

3. **DOM-base**型**XSS**是指在前端页面进行DOM操作时，带有恶意代码的片段被HTML解析、执行，从而导致XSS漏洞。

PhantomJS在XSS检测中的使用原理

PhantomJS的官网地址：http://phantomjs.org，目前最新版本 2.1。它是一个基于WebKit的服务器端JavaScript API，即在无需浏览器的支持的情况下可实现Web浏览器功能的支持，例如DOM 处理、JavaScript、CSS选择器、JSON、Canvas和可缩放矢量图形SVG等功能。基于它具有的功能，通常被用于以下场景：

1. 无需浏览器的Web测试：支持很多测试框架，如YUI Test、Jasmine、WebDriver、Capybara、QUnit、Mocha
2. 页面自动化操作：使用标准的DOM API或一些JavaScript框架（如jQuery）访问和操作Web页面。
3. 屏幕捕获：以编程方式抓起CSS、SVG和Canvas等页面内容，即可实现网络爬虫应用。构建服务端Web图形应用，如截图服务、矢量光栅图应用。
4. 网络监控：自动进行网络性能监控、跟踪页面加载情况以及将相关监控的信息

我们这里使用的主要是利用PhantomJS提供的JavaScript API 调用监控和触发接口，方便地操作html页面 DOM 节点并模拟用户操作。

在Burp Extender的BApp Store中有一个XSS的检测的插件XSS Validator，就是利用phantomJS和slimerJS的这些特性，来完成漏洞验证的。下面我们一起来看看它的原理。

在插件安装目录的xss-detector子目录下有一个xss.js的文件，就是phantomJS检测的具体实现。在代码中我们看到，默认情况下，在本地主机的8093端口启动了一个监听服务，并充当中间人代理的功能。

```
var DEBUG = true

var system = require('system');
var fs = require('fs');

// Create xss object that will be used to track XSS information
var xss = new Object();
xss.value = 0;
xss.msg = "";

// Create webserver object
var webserver = require('webserver');
server = webserver.create();

// Server config details
var host = '127.0.0.1';
var port = '8093';
```

当phantomJS服务启动，拦截到请求后即通过API接口请求页面并初始化。在初始化过程中，设置了启用web安全检测、XSS审计、js操作等。

```
// Initialize webpage to ensure that all variables are
// initialized.
var wp = reInitializeWebPage();

// Start web server and listen for requests
var service = server.listen(host + ":" + port, function(request, response) {
```

同时，自定义alert、confirm、prompt处理，记录XSS检测信息。

```
// Custom handler for alert functionality
wp.onAlert = function(msg) {
    console.log("On alert: " + msg);

    xss.value = 1;
    xss.msg += 'XSS found: alert(' + msg + ')';
};

wp.onConsoleMessage = function(msg) {
    console.log("On console.log: " + msg);

    xss.value = 1;
    xss.msg += 'XSS found: console.log(' + msg + ')';
};

wp.onConfirm = function(msg) {
    console.log("On confirm: " + msg);

    xss.value = 1;
    xss.msg += 'XSS found: confirm(' + msg + ')';
};

wp.onPrompt = function(msg) {
    console.log("On prompt: " + msg);

    xss.value = 1;
    xss.msg += 'XSS found: prompt(' + msg + ')';
};
```

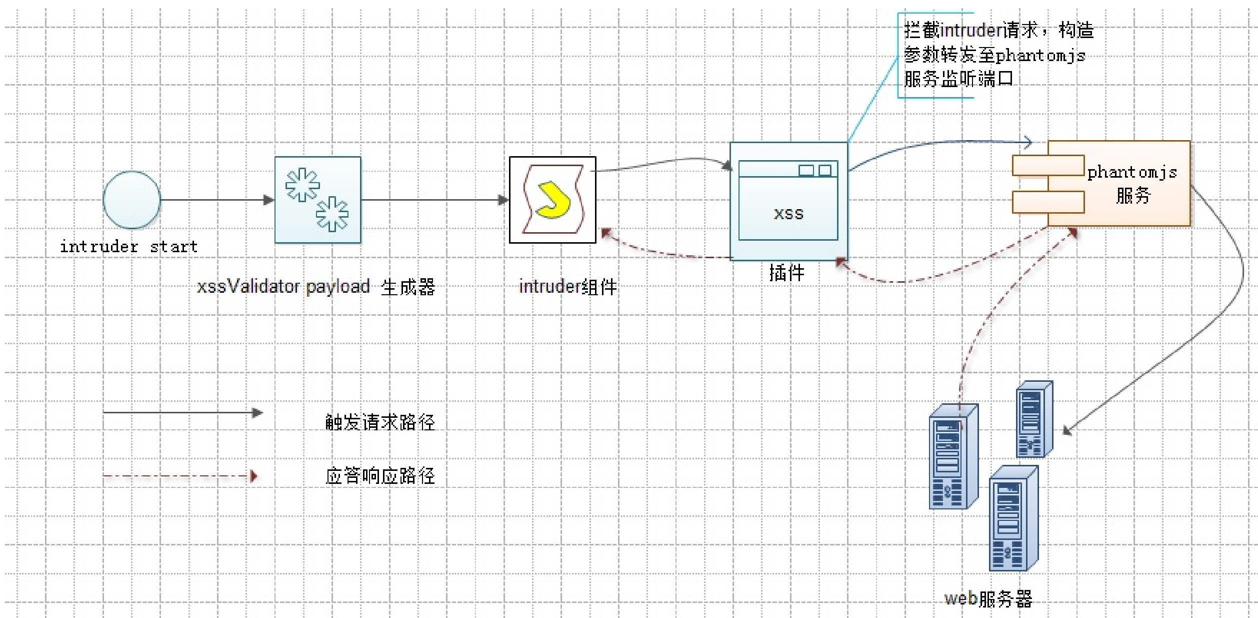
而对于js事件检测的处理，主要是通过事件分发函数去做的。

```
// Evaluate page, rendering javascript
xssInfo = wp.evaluate(function (wp) {
    var tags = ["a", "abbr", "acronym", "address", "applet", "area", "article", "aside", "audio",
    var eventHandler = ["mousemove","mouseout","mouseover"]

    // Search document for interactive HTML elements, and hover over each
    // In attempt to trigger event handlers.
    tags.forEach(function(tag) {
        currentTags = document.querySelector(tag);
        if (currentTags !== null){
            eventHandler.forEach(function(currentEvent){
             var ev = document.createEvent("MouseEvents");
                ev.initEvent(currentEvent, true, true);
                currentTags.dispatchEvent(ev);
            });
        }
    });
  // Return information from page, if necessary
  return document;
}, wp);
```

理解了这些过程，基本上XSS Validator使用phantomJS对XSS检测的原理已经掌握了。关于这个原理的类似分析，新浪微博网友@吃瓜群众-Fr1day 的文章说得很清楚，传送门地址：http://www.tuicool.com/articles/3emU7n

用图例来描述其交互过程，如下图：



在插件处理中几个关键点是需要我们特别关注的：

1. Intruder使用了XSS Validator的payload生成器，将插件与Intruder两者联动合起来。
2. 插件对Intruder发送的消息进行拦截处理，转交phantomjs服务监听端口处理。

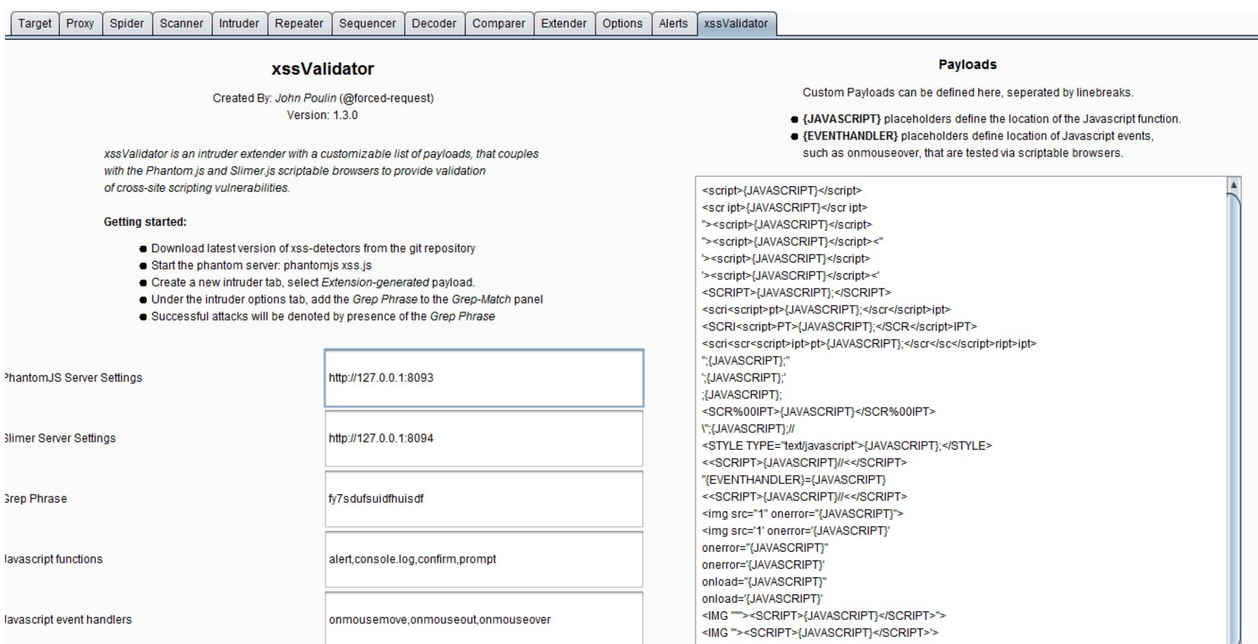
3. xss.js请求真实的web服务器，并对消息进行处理，添加Grep Phrase标志
4. Intruder组件根据Grep Phrase标志区分是否存在漏洞

只有理解了phantomJS在检测XSS中的原理，我们才可以在工作中，根据实际情况，对诸如xss.js文件进行修改，来达到满足我们自己业务需求的目的，而不仅仅拘泥了插件使用的本身功能。

使用**XSS Validator**插件进行**XSS**漏洞检测

上一节我们熟悉了phantomJS检测xss的基本原理，现在我们一起来看看XSS Validator插件的使用。

XSS Validator插件的安装依旧是可以通过BApp Store安装和手工安装两种方式，手工安装需要下载源码进行编译，这里提供项目的github地址，https://github.com/nVisium/xssValidator。安装过程由读者自己完成，如果不明白安装，请阅读Burp插件使用相关章节。安装完毕后，插件的界面如下图所示：



上图中的左侧为插件运行时需要配置的参数，右侧为验证XSS漏洞的payload。在使用插件前，有一些关于phantomjs的具体配置需要我们关注。这也是我们在通过应用商店进行插件安装时，安装界面上提供了的使用说明里的。

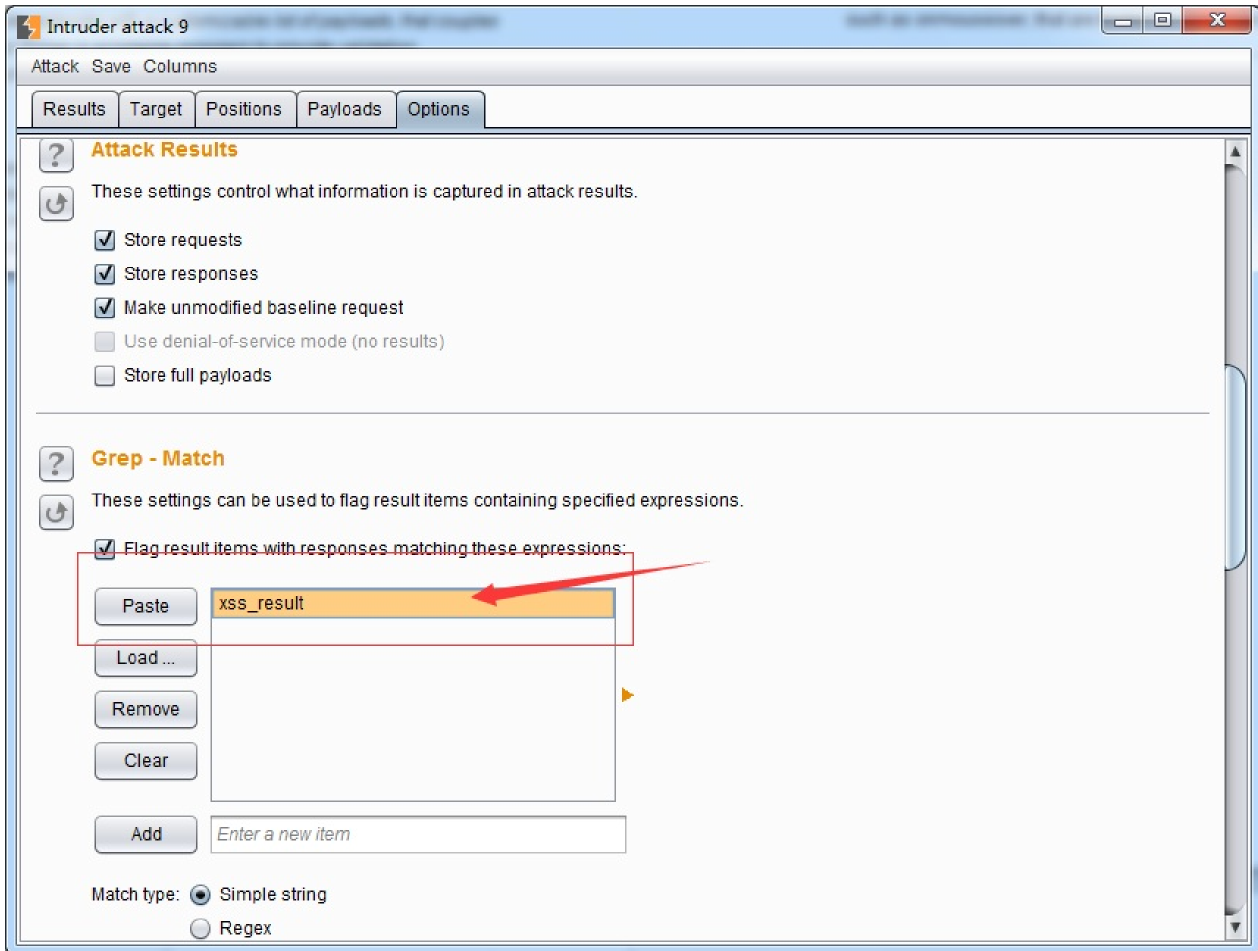
在执行Intruder之前，必须通过命令行phantomjs xss.js 启动xss检测服务，也是phantomjs的服务监听端口。这就使得我们在执行命令行之前，需要将phantomjs安装好，并加入到环境变量里，否则无法执行。至于phantomjs的安装非常简单，如果你实在不会，建议你阅读此文章。传递地址：http://www.mincoder.com/article/4795.shtml

安装完之后，执行phantomjs xss.js，控制台界面显示如下，并无其他提示信息。

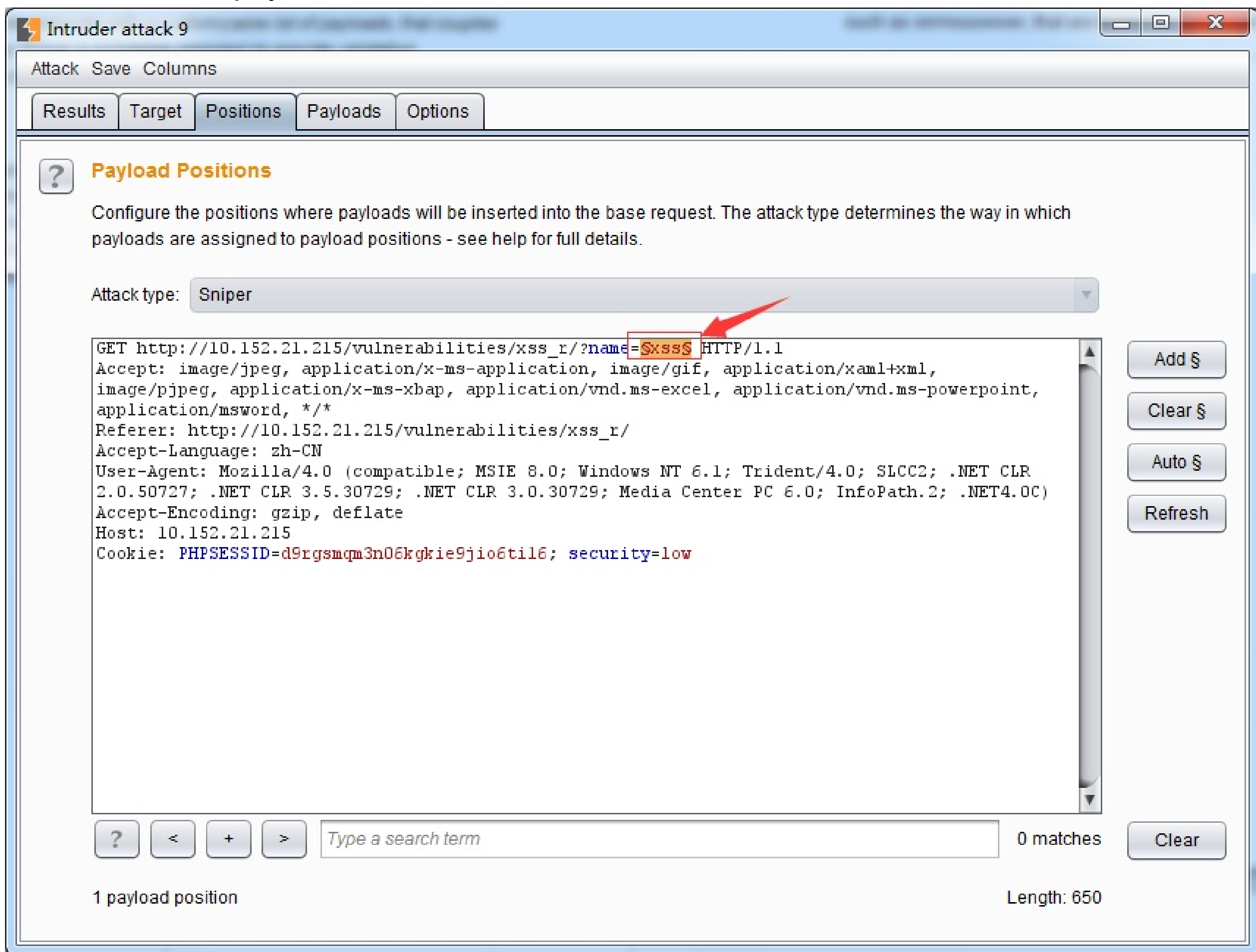
```
D:\soft\safetool\Burpsuite_pro\bapps\98275a25394a417c9480f58740c1d981\xss-detector>phantomjs xss.js
```

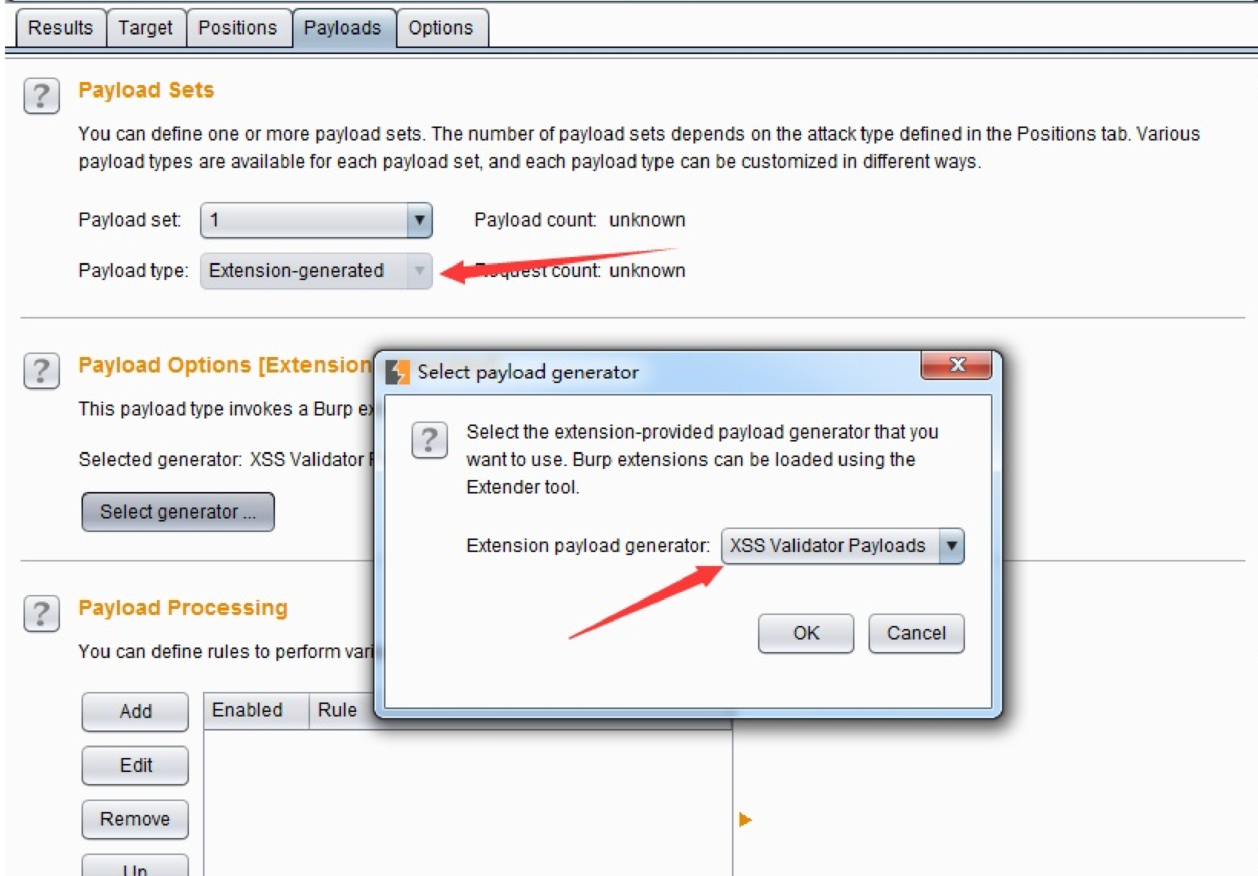
为了简单地说明使用方法，其他的参数我们都采取默认配置，只修改Grep Phrase和JavaScript functions两个参数： Grep Phrase修改为xxs_result,作为检测标志和列表头。JavaScript functions中我们仅使用alert，其他的都暂时去掉。便于我们从控制台观察结果。我们最终的配置结果如截图所示：

配置完插件之后，我们需要配置Intruder。 首先，指定Grep Phrase的值。



接着，Intruder的payload生成器需要设置为xssValidator的。





如果你如上图中所示的设置，则可以启动Intruder进行检测了。在检测过程中，我们会看到控制台输出很多日志信息，根据我们的配置，输出alert信息的表示payload检测出存在xss漏洞。如下图中2所示：

```
C:\Windows\system32\cmd.exe - phantomjs  xss.js
SyntaxError: Unexpected token '/'

  undefined:67 in setContent
  phantomjs://code/xss.js:68 in parsePage
  phantomjs://code/xss.js:183

Received request with method type: POST
Processing Post Request
Beginning to parse page
        URL: http://10.152.21.215http://10.152.21.215/vulnerabilities/xss_r/?name='''><S
CRIPT>alert(299792458)'
        Cookies: PHPSESSID=d9rgsmqm3n06kgkie9jio6ti16; security=low
SyntaxError: Unexpected EOF

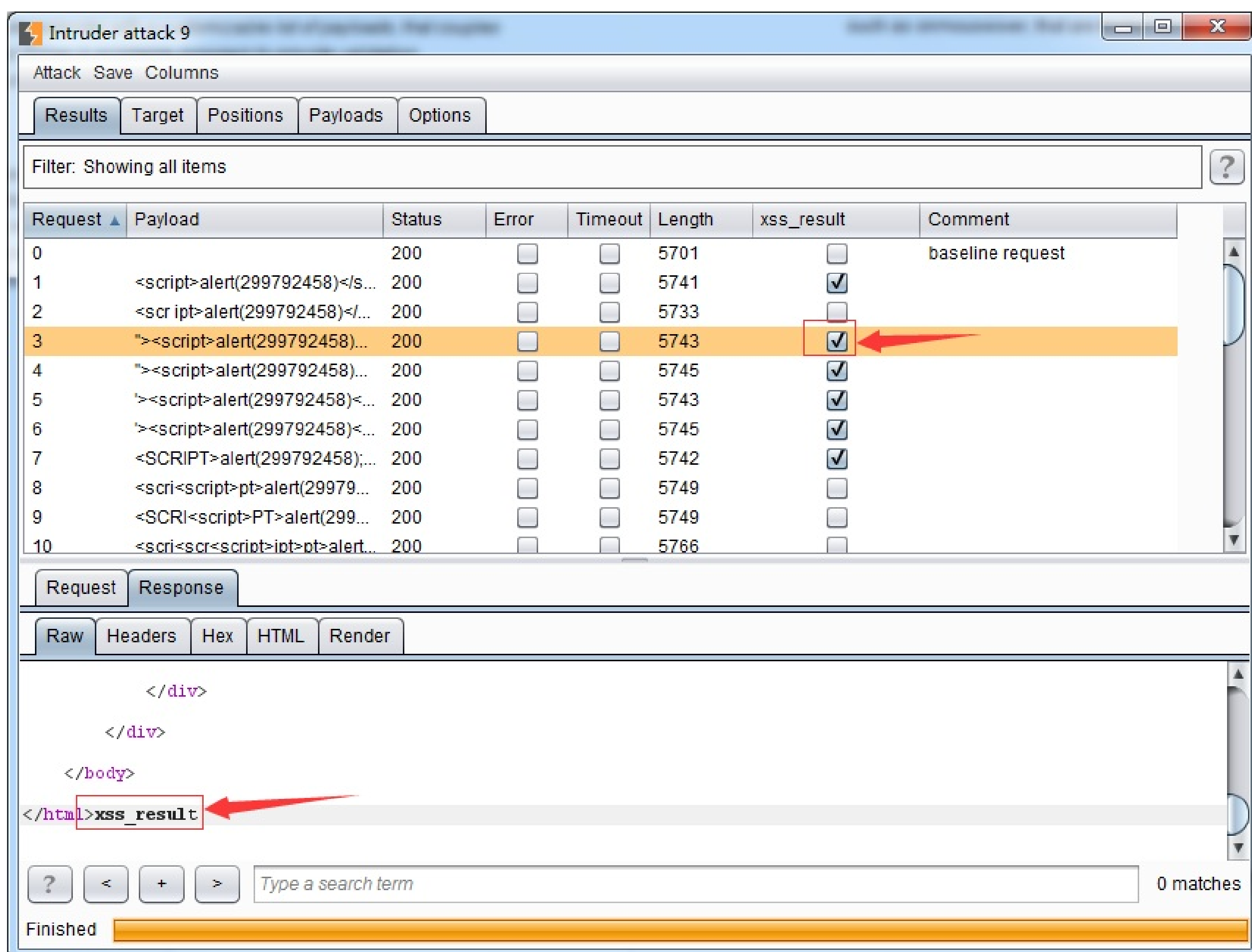
  undefined:67 in setContent
  phantomjs://code/xss.js:68 in parsePage
  phantomjs://code/xss.js:183

Received request with method type: POST
Processing Post Request
Beginning to parse page
        URL: http://10.152.21.215http://10.152.21.215/vulnerabilities/xss_r/?name=<IFRAM
E SRC='f' onerror="alert(299792458)"></IFRAME>
        Cookies: PHPSESSID=d9rgsmqm3n06kgkie9jio6ti16; security=low

Received request with method type: POST
Processing Post Request
Beginning to parse page                                     1
        URL: http://10.152.21.215http://10.152.21.215/vulnerabilities/xss_r/?name=<IFRAM
E SRC='f' onerror='alert(299792458)'></IFRAME>
        Cookies: PHPSESSID=d9rgsmqm3n06kgkie9jio6ti16; security=low

Received request with method type: POST
Processing Post Request
Beginning to parse page                                           2
        URL: http://10.152.21.215http://10.152.21.215/vulnerabilities/xss_r/?name=<IMG "
""><SCRIPT>alert(299792458)</SCRIPT>">
        Cookies: PHPSESSID=d9rgsmqm3n06kgkie9jio6ti16; security=low
On alert: 299792458
```

同时，在Intruder的执行界面上，我们可以通过xss_result来查看payload的检测情况，那些响应报文中存在漏洞标志的均被标出，便于我们对消息的区分和处理。



通过以上内容的学习，我们对PhantomJS 和xssValidator在XSS漏洞检测方面的使用有了更深入的了解。在实际应用中，由于xss漏洞的复杂性，不是靠插件默认的payload就能检测出来的，还是需要读者自己去分析和思考，找到具体的解决办法，本章内容仅仅起着抛砖引玉的作用。文章后的延伸阅读内容，感兴趣的读者可以进一步分析、实践。同时，如果有更好的此类文章，欢迎发邮件给我t0data@hotmail.com，我会添加到延伸阅读里。

> 延伸阅读：1.Server-Side-XSS-Attack-Detection-with-ModSecurity-and-PhantomJS

2.如何使用开源组件解决web应用中的XSS漏洞

第二十章 使用Burp 、Android Killer进行安卓app渗透测试

很多人发邮件询问这章什么时候写，在此统一回应一下

平时工作较忙，俗务缠身，暂时没有写的计划

网上相关的文章比较多，可以参考一下

https://www.secpulse.com/archives/4325.html

http://www.freebuf.com/articles/web/29421.html

https://zhuanlan.zhihu.com/p/22302904